画素形漢字パターンの出力処理に関する研究

1982年4月

森　克己

画素形漢字パターンの出力処理に関する研究

１９８２年４月

森　　克己

目　　次

第1章　序　　論

近年，漢字コード表のＪＩＳ制定[(1)]とあいまって，企業での事務処理分野におけるオフィスオートメーションや文書処理システム，また，一般家庭まで対象に含めた情報サービス分野におけるキャプテンシステムや画像応答システムなど，日本語情報を取り扱う各種のシステムの社会への導入が急速に進んできた。

このような日本語情報処理システムでは，我が国の国字である漢字の入出力技術が重要となる。

漢字は字種が多く，かつ，字形が複雑であるため，従来の英字，数字のみを対象とした場合に比較すると多くの困難な問題を引起す。

これらの問題のうち，漢字の入力に関する問題は早くから指摘され数多くの研究が行われてきた。[(2), (3)]しかし，漢字の出力に関しては経済性を無視すれば，人間が読むことができる漢字情報を出力するという，最低限の機能は実現できたため，顧みられることは少なかった。

しかし，多くの日本語情報処理システムでの漢字の出力機能は，わずか１種類の書体の低品質の漢字パターンを，しかも，固定された大きさで表示，記録できるにすぎないのが現状である。

このような漢字の出力技術上の問題点も，前述のような日本語情報処理システムの普及とともに認識されはじめてきた。すなわち，現在の優れた活字文化の中にあっては，従来の日本語情報処理システムの出力である文書や画面は表現能力や品質の点で劣っているといわざるを得ない。

優れた活字文化が定着した中で，今後，計算機を用いた日本語情報処理システムが，さらに普及してゆくためには，字種，書体，サイズさらに字体の選択の融通性に富んだ漢字出力技術の確立が不可欠であると思われる。

本論文では，以上のような背景のもとで，画素形漢字パターンを対象として，漢字パターンの出力機能の拡充を目的として行った研究結果を述べるものである。

1.1 漢字パターン処理の研究と本論文の位置および概要

漢字パターン処理を入力用処理と出力用処理とに分けて考えるとき，入力用漢字パターン処理の研究は漢字認識法の研究として昭和３０年代の後半から開始されており，長い歴史をもっている。

一方，本論文が対象とする出力用漢字パターン処理の研究は１９６０年代の後半から開始され，１９７０年代後半に入って多くの研究報告が見られるようになった。[4]

図 1.1 に本論文で扱かう漢字パターンのデータ圧縮，漢字パターンのサイズ変換，書体変換，それに漢字パターンの作成の各処理に関する研究の経過を示す。

(1) データ圧縮処理の研究

出力用漢字パターン処理の主要な目的は経済的な漢字パターンの発生技術を確立することにある。そこで，まず，出力用漢字パターン処理の研究は，大量の漢字パターンの蓄積に必要となるメモリの使用量を削減することを目的とした漢字パターンのデータ圧縮法の研究から開始された。

漢字パターンの表現法としてはストローク方式と画素方式に分かれるが，まずストローク方式についてみると，１９６９年に坂井，長尾，寺井によって“部分パターンによる漢字の合成方式”が発表された。[4-1]これは漢字を構成する偏や旁などの部首を部分パターンとして，それらを，大きさ，位置を変えて適宜組み合わせることにより漢字パターンを記述する方法である。比較的少数の部分パターンの記述データと，各漢字パターン毎に部分パターンの組み合わせ方を示すオペレータとによって漢字パターンを記述できることを示している。また，その結果として漢字パターンのデータ量の削減効果を得ている。

この研究で示された，部分パターンの組み合せで漢字パターンを表現する考え方は，その後，入・出力技術の両方に引継がれた。出力技術としては，さらに部分パターンの種類とそれらの組み合せ方を示す記述フレーム数を増やして，合成文字品質を向上させる研究[5], [6]へと進み，入力技術としては，偏，旁などを示す部首キーの複数回打鍵入力方式の研究が報告されている。[5]

また，線幅を１画素として“筆”の平面的な動きを示す二次元情報のみを符号化する方法[7], [8], [9]に加えて，漢字パターンの高品質化の要求に応えて，線幅をもった漢字パターンのストローク表現法の研究も報告されている。[10], [11], [12], [13]これは，後述する漢字パターンの書体変換処理へと発展する動きを含んでいる。

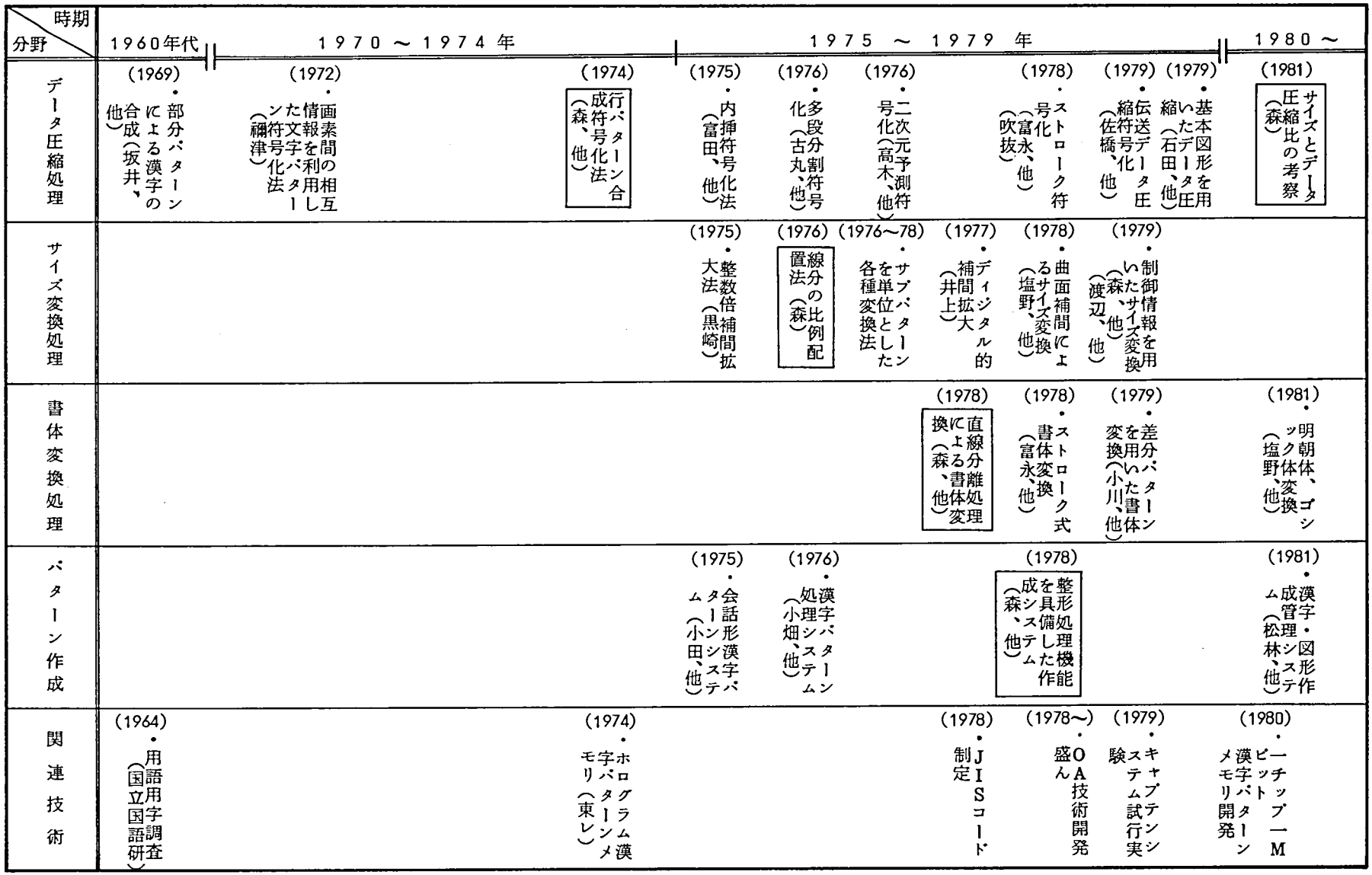


図 1.1　漢字パターン出力処理の研究経過

一方，画素方式はＣＲＴ表示装置やファクシミリ受信機などラスタ走査形出力装置との適合性が良く，出力装置の選択範囲がストローク方式に比較して優れているため，１９７０年代の前，中期を中心に画素形漢字パターンを対象とした研究が多く見られるようになった。

まず，１９７２年に禰津によって“画素間の相互情報量を利用した文字パターンの符合化法”が発表された。この方法は画素形漢字パターンを（２×２）画素から成るサブパターンに分割し，サブパターンがとり得る１６通りの状態を，その１次マルコフ過程としての遷移確立に基づいて符号化することによってデータ量を削減している。この結果，近似を含まず（２４×２４）の漢字パターンに対して５２.５％と云う，現在までに報告された最も高い圧縮効率を達成している。(4−2), (14)

さらに，近似を含まない画素形漢字パターンの符号化法としては，多段分割符号化方式(15)や二次元予測符号化方式(16)などが発表されている。

また，近似符号化法としては，縦，横方向に１画素おきにサンプリングした画素のみを符号化し，それ以外の画素値は内挿することにより（３２×３２）の漢字パターンに対してデータ量を3/8に圧縮した報告がなされている。(17)

これらの研究報告から（２４×２４）あるいは（３２×３２）程度の漢字パターンに対してはデータ圧縮効率の限界が1/2程度であることが明らかになったこと，また，外部要因としてメモリ価格が急速に低下してきたことから，１９７０年代の後半以降，（３２×３２）程度以下の低品質の漢字パターンを対象としたデータ圧縮法の研究は下火となった。

しかし，出力表示装置の高解像度化と漢字パターンの種類の多様化への要求に応じて，画素数が（６４×６４）～（１２８×１２８）にも達する高品質な漢字パターンに対するデータ圧縮処理の研究が，画素配列法の考察まで含めて行なわれるようになってきた。(15), (18), (19)

さらに，漢字パターン発生器をもたない，簡易な多数の漢字出力端末を含む文字情報提供システムのための漢字パターンの伝送データ量の圧縮法の研究が報告されている。(20), (21), (22)

著者は，多数のファクシミリ受信機を端末装置として，センターに蓄積された文字，図形情報をそれらの端末装置に出力する，ファクシミリ応答システム(23)用の漢字パターン発生器の経済化を目的に１９７２年から画素形漢字パターンのデータ圧縮法の研究を開始し，１９７４年に，部分パターンの形状を１次元パターンに限定した“行パターン合成符号化法”を提案した。(24), (25)

この符号化法は，漢字パターンは極めて高価であるため，多数の端末装置で共用するこ

とを前提条件として開発したものであり，端末装置の走査方向と一致する１次元部分パターンを用いることにより，前記の条件を満足し，かつ，データ量を（２４×２４）漢字パターンについて約 2/3 に圧縮した。

さらに，７０年代の後期に入り，実用的な漢字処理システムの出現とともに，発生できる漢字パターンの種類の拡充が要求されるようになってきた。[26] しかし，７０年代の前，中期を中心に行なわれたデータ圧縮法の研究では，漢字パターンの画素数は高々（３２×３２）程度であり，それ以上に画素数の多い高品質の漢字パターンに対するデータ圧縮効率の定量的考察は行なわれていなかった。

そこで，画素数が異なる低品質から高品質まで，複数種類の漢字パターンを含むシステム設計の基礎資料を得ることを目的に，画素数とデータ圧縮比の関係について研究を開始し，定量的なデータ圧縮比と画素数の関係を明らかにした。[19], [27]

(2) サイズ変換処理

漢字情報システムの実用期に入ってくるとともに，図，表を含んだ文書編集など，漢字情報処理システムのソフトウェア機能の充実と，漢字ディスプレイ装置，漢字プリンタなど解像度の異なる種々の出力装置の出現によるハードウェア機能の充実を背景として，漢字パターンのサイズや形状をきめ細かく制御したいという要求が強くなってきた。

従来の漢字パターン発生技術では，発生すべき漢字パターンは全て予じめ漢字パターンメモリ内に蓄積しておくことが必要であったため，この要求に対して応えることは出来なかった。

画素形漢字パターンのサイズ変換処理については１９７５年に発表された黒崎の研究[4-3]がある。しかし，この方法は画素のサイズを単純に整数倍に拡大し，斜線部を補間・平滑化するものであるため，サイズの変換は整数倍の拡大に限られていた。

したがって，縮小まで含めた非整数倍のサイズ変換処理については本研究以前には見当らない。本論文ではサイズ変換処理に対する要求条件と基本的な問題点を明確にするとともに変換アルゴリズムを考察する。

漢字パターンが縦，横直線を多数含んでいるという特徴を利用して，文字バランスと縦，横直線幅の統一という２つの主要な文字品質の決定要因を制御し得るサイズ変換手法を“線分の比例配置法”として１９７７年に提案した。[28] この方法は，さらに，著者らと渡部らによって，ほぼ同時に高速化へと改良され，[29], [30] 実用装置へ組み込まれた例が報告されている。[31]

その後，サイズ変換処理は，漢字パターンを一般の２値図形と見なしてサブパターンに分割し，サブパターンを単位に変換テーブルを用いて変換する方法[32],[33]や画素単位に周囲画素との間で論理演算処理を行ない，画素の挿入・削除の可否を判定する方法[34],[35][36]漢字パターンを多値パターンと見なし，変換後の画素数に対応する密度でサンプリングを行ない，そのサンプリングポイントの濃度を周囲の画素値から曲面補間によって求め２値化する方法[37]などが１９７７年から７９年にかけて多く発表されるに至った。

(3) 書体変換処理

書体変換処理の研究の目的はサイズ変換処理の目的と同じく，経済的に多種類の漢字パターンを発生することにある。

画素形漢字パターンの書体変換処理に関する研究は本研究以前には見当らない。

本論文では（３２×３２）の明朝体とゴシック体を対象として両書体間の変換法について考察し，直線幅と“ウロコ”等の線端飾りの有無を制御することにより，擬似的な書体変換が可能であることを示し，残された課題を明らかとした。

本研究以降，原漢字パターンと変換後の漢字パターンとの差分パターンを符号化することにより，サイズと書体の両変換を行なう差分パターン法[38],[39],[40]や，明朝体からゴシック体への変換処理の研究[41],[42]が１９７９年から８１年にかけて見られる。

また，ストローク形漢字パターンの書体変換としては富永らの報告がある。[10],[43],[44]

(4) 漢字パターン作成処理の研究

従来，漢字パターンはデザイナーらの人手によって作成されていたため，その作成には多大の労力と経費と時間を要していた。

計算機を用いた効率的な漢字パターン作成法の研究は１９７５年の，小畑，福森，金出による，計算機との対話処理による乗車券用駅名文字パターンの作成システムの報告[45]を初めとして，同様のシステムの報告[46],[47]が見られる。

これらのシステムでは，文字母形をサンプリングし，ディジタル文字パターンを得た後，そのパターンに対して人間が対話的に修正作成する方式をとっている。これにより，従来の人手による作成法に比較して作成効率の大幅な向上が達成された。

しかし，文字母形をサンプリングして得られるディジタルパターンは，直線周辺に多くの凸凹（以後，ノッチと呼ぶ）や線幅の不揃いを含んでおり，これらの比較的単純な歪みの修正に会話修正時間の多くを費やしていた。

本研究では，ノッチと線幅の不揃いを除去する整形処理法を検討し，漢字パターン作成

装置に付加するとともに，装置構成法として，複数個の整形処理結果の中から最適結果をオペレータが選択する会話方式を導入することにより，オペレータの修正作業量の軽減と質的改善を図った。

1.2 研究の目的と論文の構成

本研究の目的は漢字パターン発生器の経済化と多機能化にある。

図1.2に漢字情報システムにおける漢字パターン発生器の位置と本研究が対象とする範囲を示す。

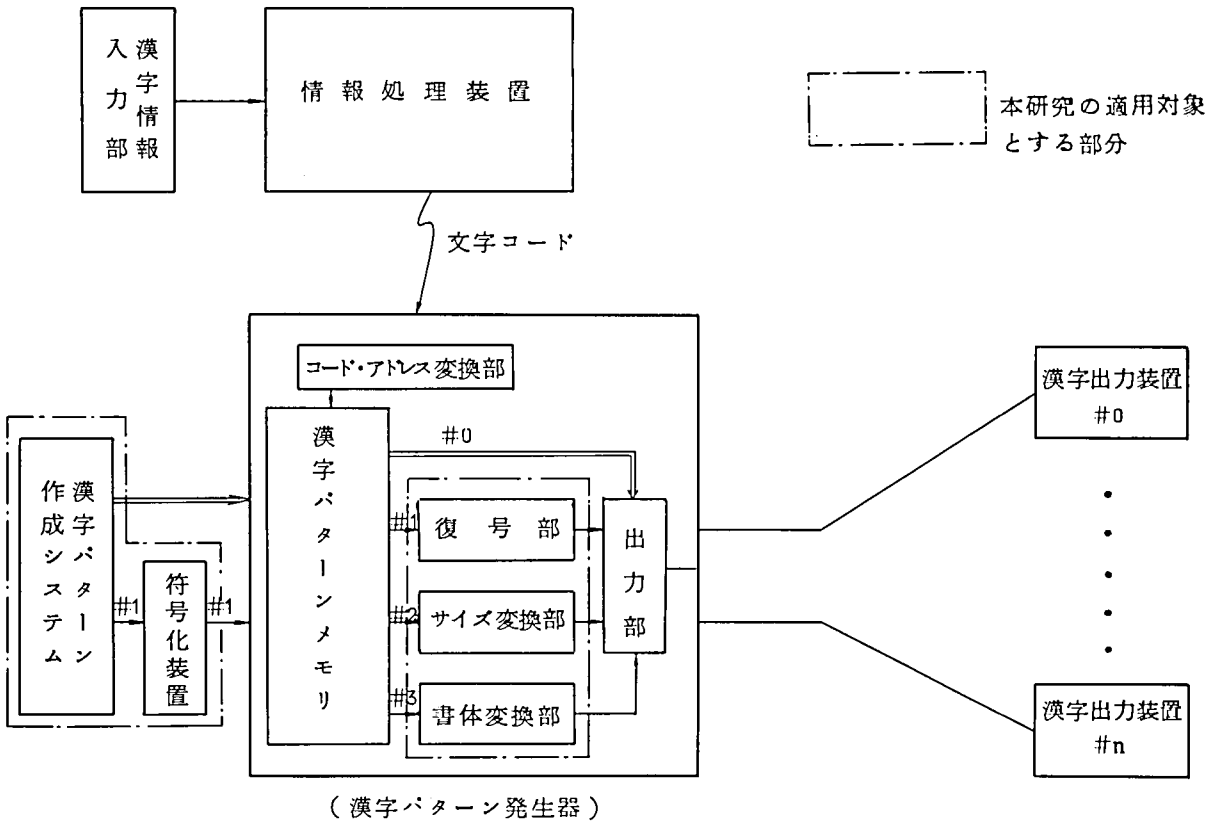


図1.2　漢字情報システムの構成と研究の位置づけ

漢字パターン発生器は情報処理装置から送られてくる漢字コードを人間が理解できる漢字パターンへと変換する機能をもつ装置であり，基本的には漢字パターンを記憶する漢字パターンメモリと漢字コードをパターンメモリのアドレスに変換するコード・アドレス変

換部から構成される。

漢字パターンメモリ[48]としては特殊電子管，光学フィルム，電子メモリなどがあり，それぞれ図1.3に示すような漢字パターン発生方式に分類される。

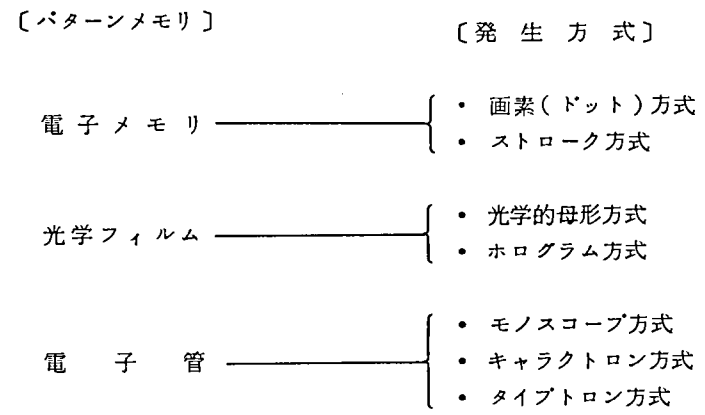


図1.3　文字パターン発生方式

本研究では電子計算機との整合性の良い電子メモリを用いる方式の中で，特に，出力表示装置の選択の範囲が広い画素方式を対象とする。

画素方式は漢字パターンを白，黒(あるいは0，1)の2つの状態をもつ画素の集合として表現し，記憶する方式であり，画素数を変えることにより複雑な漢字パターンを記憶することができる融通性のある方式である。しかし，表現に用いた画素数がそのままパターンの記憶に要するメモリ量となるので，大容量のパターンメモリが必要となる。

したがって，画素数の多い高品質の漢字パターンを発生させる場合や，サイズや書体の異なる複数種類の漢字パターンを発生させる場合には漢字パターン発生器が極めて高価になってしまうという問題がある。この問題はメモリ価格が急速に低価した現在でも解決されておらず，1つのシステム内で発生可能な漢字パターンは，せいぜい，画素数が(32×32)程度以下で，種類も1種類の場合が大多数である。

これらの状況を背景として，本研究では漢字パターン発生器の経済化と高機能化を目的として以下の項目について考察する。

(1) 漢字パターンデータの圧縮

漢字パターンメモリの所要量を削減することを目的とした，漢字パターンデータの圧縮法の研究。

図1.2の漢字パターン発生器において，漢字パターンメモリ内に符号化されて蓄積され

た漢字パターンデータは＃１のパスを通り復号されて出力部へ送られる構成となる。

(2) 漢字パターンのサイズ変換

基準となる１種類の漢字パターンから，構成する画素数を自動的に変えることによって，パターンメモリの量を増やすことなく，複数種類の漢字パターンを発生するサイズ変換法の研究。

図1.2の漢字パターン発生器において，漢字パターン内のパターンデータは＃２のパスでサイズ変換部を経て出力部へ送られる構成となる。

(3) 漢字パターンの書体変換

基準となる１つの書体の漢字パターンから，自動的に書体を変えることにより，パターンメモリの量を増やすことなく，異なる書体の漢字パターンを発生する書体変換法の研究。

図1.2の漢字パターン発生器において，漢字パターンメモリ内に蓄積された漢字パターンデータは＃３のパスで書体変換部を経て出力部へ送られる構成となる。

(4) 漢字パターンの作成

効率的な漢字パターンの作成のための自動整形処理法と会話処理方式に関する研究。

これらの４つの研究項目は互いに独立ではない。データ圧縮により，少ないパターンメモリ量で発生すべき全ての種類の漢字パターンが蓄積できれば，サイズ変換，書体変換の必要性は無くなる。また，変換処理により品質の良い漢字パターンが得られれば，データ圧縮および漢字パターン作成技術の必要性は小さくなる。

さらにまた，サイズ変換，書体変換処理は漢字パターンデータ圧縮処理と作成処理の一部として位置づけることもできる。

以下に本論文の構成を示す。

２章では漢字パターンデータの圧縮について述べる。データ圧縮処理はその復号処理と独立して考えることは出来ない。ここでは，１台の漢字パターン発生器を多数のラスタ走査形出力装置で共用する条件で，ラスタ走査形出力装置に適したデータ圧縮法として検討した“行パターン合成符号化方式”について最適化と符号化効率について述べる。最後に，漢字パターンの構成画素数とデータ圧縮比の関係について定量的な考察を行なう。

３章では画素形漢字パターンのサイズ変換処理について述べる。まず，サイズ変換処理での問題点を明らかにし，それを解決する変換法として“線分の比例配置法”について原理と処理結果を示した後，処理の高速化手法について述べる。最後に適用領域について考

察する。

4章では漢字パターンの書体変換処理について述べる。すなわち，（３２×３２）の明朝体とゴシック体の漢字パターンを対象に，書体変換を行なうために制御すべき書体間差分を抽出し，それらの書体間差分の制御の可能性について考察する。

5章では人間と計算機との会話処理による漢字パターンの作成について述べる。特に，自動整形処理とその処理機能をパターン作成装置へ組み込んだ時の合理的な会話処理方式について考察する。

6章では本研究の成果をまとめるとともに，今後に残された課題を整理する。

第2章　漢字パターンデータの圧縮

漢字パターンデータの圧縮処理の対象分野としては，漢字パターンメモリの容量の削減を目的とする場合とパターンデータの伝送量の削減を目的とする場合がある。両者の基本的な相違点は漢字の使用頻度を考慮するか，否かにある。

本論文では，以下，漢字パターンメモリ容量の削減を目的に，漢字の使用頻度の偏りを考慮しない立場でのデータ圧縮処理について考察する。

2.1　ま　え　が　き

電子的な漢字パターン発生方式では発生すべき漢字パターンを電子的なメモリ装置内に蓄積しておくことが必要である。

この時必要となるパターンメモリの容量は蓄積すべき，すなわち，発生すべき漢字の字種数と漢字パターンの品質，すなわち，1個の漢字パターンの表現に使用されるデータ量の積となる。例えば，（３２×３２）画素数の漢字パターンを，ＪＩＳ第２水準まで含めて6,５００字発生させるためには８００Ｋバイト強のパターンメモリが必要となる。

従来，このパターンメモリ容量の大きさ，すなわち，漢字パターンを表現するデータ量の多さが，漢字パターン発生器を高価なものとし，漢字情報を出力する際の大きなネックとされてきた。

この漢字パターン発生器の経済化は，パターンメモリ素子の経済化と所要メモリ量の削減の２通りの方法で達成される。

ＩＣメモリ素子の低価格化は集積回路技術の進歩に支えられて急速に進んできており，（１６×１６）画素の漢字パターンなら1チップで約4,０００字を蓄積し得る1メガビット容量の漢字パターン用のマスクＲＯＭも開発されつつあり，[49]漢字パターン発生器の経済化を大きく進めることが期待される。

しかし，漢字パターン発生器の適用面から考えてみると，大規模な漢字情報処理システムにおいては，画素数にして（１６×１６）程度の低品質の漢字パターンから（１２８×１２８）程度の極めて高品質の漢字パターンまで，数段階の異なる画素数の漢字パターン

を，明朝体，ゴシック体など複数の書体について発生させ得ることが要求される。

例えば，画素数が（１６×１６），（３２×３２），（６４×６４），（128×128）の４種類の漢字パターンを6,５００字種ずつ，明朝体とゴシック体の２書体について発生させる場合では，約３５メガバイトのパターンデータ量となり，システム構成上の大きな負担となる。

一方，小規模の漢字出力装置を考えると，漢字パターンの発生速度は漢字プリンタの速度あるいはディスプレイ画面上では人が読む程度の速度であれば良く，ＩＣ－ＲＯＭのパターンメモリで達成される数万字／秒に達する発生速度[49]は必要とされない。

発生速度的には現存するワイヤドット式，感熱式，静電式あるいはインクジェト式の漢字プリンタ装置では数１０字／秒から高々数１００字／秒程度の発生速度が達成できれば良く[2-1]こゝでも，やはり，発生可能な漢字パターンの種類の多さに対する要求が強いと考えられる。

以上述べたように，ＩＣメモリコストの急速な低下にもかかわらず，必ずしも，漢字パターンの出力上の問題は解決されてはいない。

そこで，メモリ価格の急速な低下という技術の流れの中にあっても，上記の問題の解決法として漢字パターンデータの圧縮技術が必要となる。

大規模システムにおいては上記の問題の方式的解決法として，字種の使用頻度の偏りに基づく，パターンメモリの階層構成法が有効であると考えられる。[5]すなわち，使用頻度の高い字種の漢字パターンデータだけを高速なＩＣメモリ内に格納しておき，低使用頻度の漢字パターンデータは低速の大容量ファイルに格納しておき，必要な都度，高速メモリ上へ読み出す方式である。

この方式の選択の可否は，データ圧縮処理によるパターンメモリ量の削減効果と低速大容量ファイル装置を使用することによる漢字パターンの発生速度の低下とのかね合いで決まることになる。

本章では，従来のデータ圧縮処理の研究が高々（３２×３２）程度以下の低，中品質の漢字パターンを対象に行なわれ，画素数の大きな高品質漢字パターンまで含む，大規模システムの設計には不充分なデータしか無かったことを背景として，大規模システム設計のための基礎資料を得ることを目的として，画素形漢字パターンのサイズとデータ圧縮比の関係について定量的に考察する。

一方，簡易な出力装置を含む小規模システムにおいては，発生速度に対する緩やかな条

件から生じる時間的な余裕を漢字パターン発生器の経済化へ転化することが必要であり，そこにデータ圧縮技術が適用され得る。すなわち，圧縮処理の適用により漢字パターンの発生速度が1～2桁下っても，データ量を1/2～1/3に削減できれば，同一価格で2～3種類の異なる種類の漢字パターンの発生が可能となり機能的に大幅に向上させることが出来る。

しかし，出力装置対応に漢字パターン発生器を持てない時点では，1台の漢字パターン発生器を多数の出力装置で共用する方式が現実的なシステム構成方式としてとられる。この方式の例としては電電公社で検討を進めてきた画像応答システム（ＶＲＳ），[50]ファクシミリ応答装置（ＦＲＥ），[51]それに１９７９年に郵政省を中心に試行サービスが開始されたキャプテンシステム[52]があげられる。

本論文で考察する“行パターン合成符号化方式”は，多数のファクシミリ受信機を端末装置として，各端末に異なる文字，図形情報を出力するファクシミリ応答装置用漢字パターン発生器を対象として検討したものである。

多数の出力装置で時分割で1個の漢字パターン発生器を使用する場合，漢字パターンデータの符号化，復号化の処理は，1割当て時間内に出力装置に出力できるパターンを単位として行なうことが望ましい。

従来報告されているデータ圧縮処理は，符号化，復号化は二次元的な部分パターンを単位としたものであり，ラスタ走査形出力装置で共用する漢字パターン発生器へ適用することは不適当であった。

もし，複数の走査線にまたがる部分を単位として漢字パターンが発生されるならば，最悪の場合端末装置の横方向の表示文字数に端末装置の台数を乗じた文字数について，部分パターンを蓄積するメモリをシステムは持たなければならなくなり，新たなメモリ増加と複雑なシステム構成となってしまう。

そこで，部分パターンの形状を走査線方向と方向が一致する一次元パターンに限定して，圧縮効率の有無について検討する。

部分パターンを単位とした符号化処理では部分パターンの形状と大きさがデータ圧縮効率を左右する。そこで，部分パターンの形状を一次元に限定した上で，特に従来の研究において検討されていない，部分パターンの大きさに対する考察を行ない最適な大きさがあることを示す。

2.2 行パターン合成符号化法[25],[53]

データ圧縮処理は当然その復号過程と独立して考えることはできない。圧縮法は復号過程まで含めて総合的に評価する必要がある。

現在，大きなデータ量を伴ない，それだけデータ圧縮処理の必要性が高い精細な漢字パターンを，高い品質で表示できる装置としてはファクシミリ受信機やＣＲＴ表示装置などいわゆるラスタ走査形表示装置が主流をなしている。これらのラスタ走査形表示装置上への漢字パターンの表示を行なう場合は，各走査線上に表示される部分パターンを単位として符号化，そして復号されることが望ましい。特にセンタ・ツ・エンド形の情報提供システムなどでの適用において，１台の漢字パターン発生器を複数の表示装置で時分割により共用する場合には，部分パターンの発生は表示装置の走査線ごとに完結することが望ましい。

そこで，本節では部分パターンの形状を表示装置の走査方向と方向が一致する一次元パターンに限定して，圧縮効率の面からみた部分パターンの最適な大きさを求め，漢字パターンデータの圧縮効果の有無について考察する。

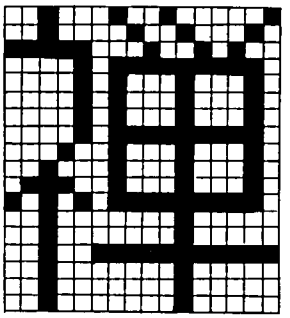
2.2.1 行パターン合成符号化法の原理

考察の対象とする漢字パターンは図 2.1 に示したように，横方向に n 画素，縦方向に m 画素が配置された（ m 行×n 列）の画素形漢字パターンである。

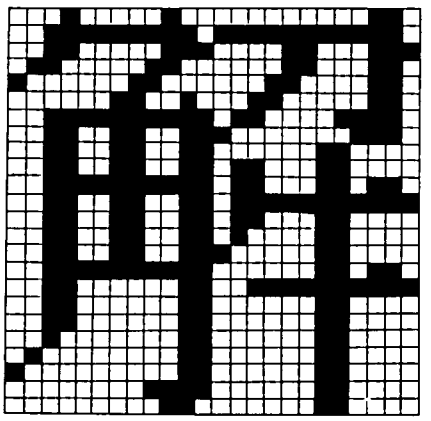
漢字パターンを形成する，横方向の（１行×n 列）の画素の並びを行パターン，行パターン内の画素の個数 n を行パターンの長さとよぶこととする。

このとき，（ m × n ）の漢字パターンＡは列方向に m 個の行パターン B_i を合成したものと見なすことができる。このことを次のように書くこととする。

$$A = (B_1, B_2, \cdots\cdots, B_m) \qquad (2.1)$$



(a)（18×16）



(b)（24×24）

図 2.1　画素形漢字パターンの例

この時，m 個の行パターン B_i（$i=1, 2 \cdots\cdots m$）は漢字パターンの形状によって，すべて異なる場合もあるし，また，複数個が一致する場合も有り得る。

次に，考察の対象を各個別の漢字パターンから，漢字パターン発生装置が発生すべき漢字パターン全体の集合 $\{A^j\}$ に広げる。

各漢字パターン A^j は式（2.1）と同様に

$$A^j = \{ B^j_1, B^j_2, \cdots\cdots, B^j_m \} \qquad (2.2)$$

と表わされる。

漢字パターンの集合 $\{A^j\}$ の大きさとしては数千から1万字程度を含む場合を考える。

こうして得られた行パターンの集合 $\{B^j_i\}$ の中には多数の同一の行パターンが含まれることが予想される。各行パターン B^j_i が行パターンの集合 $\{B^j_i\}$ の中で有する同一行パターンの個数を行パターン B^j_i の出現頻度とよぶこととする。

次に，行パターンの集合 $\{B^j_i\}$ の中から同一の行パターンはその中の1つのみを残し他は取り除く操作により，すべて異なった行パターンだけから成る集合 $\{B_k\}$ を作る。このすべて異なった行パターンのみからなる行パターンの集合 $\{B_k\}$ を基本行パターンの集合，その要素 B_k を基本行パターンとよぶ。

さらに，基本行パターン B_k と1対1に対応する符号 C_k を要素とする符号集合 $\{C_k\}$ を考える。

以上の操作の結果，漢字パターンの集合 $\{A^j\}$ から基本行パターンの集合 $\{B_k\}$ と符号集合 $\{C_k\}$ を作ることができた。

漢字パターン A^j を構成する行パターン B^j_i は必ず基本行パターンの集合 $\{B_k\}$ に含まれ，したがって対応する符号も必ず符号集合 $\{C_k\}$ の中に存在することになる。

この結果，漢字パターン A^j は m 個の符号の組として

$$A^j = \{ C^j_1, C^j_2, \cdots\cdots, C^j_m \} \qquad (2.3)$$

$$C^j_i \in \{C_k\}, \quad i = 1, 2, \cdots\cdots, m$$

と表わすことができる。

基本行パターンの集合 $\{B_k\}$ の大きさ（要素である基本行パターンの個数）は行パターンの長さが n であれば高々 2^n であり，実際には 2^n よりはるかに小さくなる。そ

の結果，適当な符号集合 $\{C_k\}$ を用いることによって漢字パターンデータの圧縮効果を得ることが期待できる。

以上述べた方法で漢字パターンの集合 $\{A^j\}$ を記憶するために必要となるデータ量 Q は，基本行パターンの集合 $\{B_k\}$ としてのデータ量 Q_1 と式（2.3）に従って，各漢字パターン毎に基本行パターンの組み合わせ方を示す符号の組を表わすデータ量 Q_2 の和として

$$Q = Q_1 + Q_2 \qquad (2.4)$$

となる。

この時，データ圧縮比 ε は

$$\varepsilon = \frac{m \cdot n \cdot P}{Q} \qquad (2.5)$$

で与えられる。ここで P は漢字パターンの集合 $\{A^j\}$ の大きさ，すなわち，符号化の対象とする漢字の個数である。

以後，考察を行なう便宜上，圧縮比の逆数として圧縮率 η，η_1，η_2 を以下に定義する。

$$\left.\begin{aligned} \eta &= \frac{1}{\varepsilon} = \eta_1 + \eta_2 \\ \eta_1 &= \frac{Q_1}{m \cdot n \cdot P} \quad , \quad \eta_2 = \frac{Q_2}{m \cdot n \cdot P} \end{aligned}\right\} \qquad (2.6)$$

上記の方法を以後，“行パターン合成符号化法”と呼ぶことにする。

以上，行パターンの長さを n として，行パターン合成符号化法の原理について述べた。

以下，行パターンの分割を行なうことを考える。行パターンの長さ n は必要な数に分割される場合を考える。

分割をしない長さ n の行パターンを 1 行パターン，2 等分割して長さが $n/2$ のものを 1/2 行パターン，4 等分割して長さが $n/4$ のものを 1/4 行パターンと呼ぶことにする。

2.2.2 基本行パターンの集合 $\{B_k\}$ の統計的性質

行パターン合成符号化法のデータ圧縮効率は基本行パターンの集合 $\{B_k\}$ の大きさと各基本行パターンの出現ひん度により決定される。

そこで，（１８行×１６列）のゴシック体漢字パターンと（２４行×２４列）の明朝体漢字パターンを対象として，行パターンの分割数をパラメータに，漢字パターンの集合 $\{A^j\}$ の大きさと基本行パターンの集合 $\{B_k\}$ の大きさの関係を図2.2に示す。

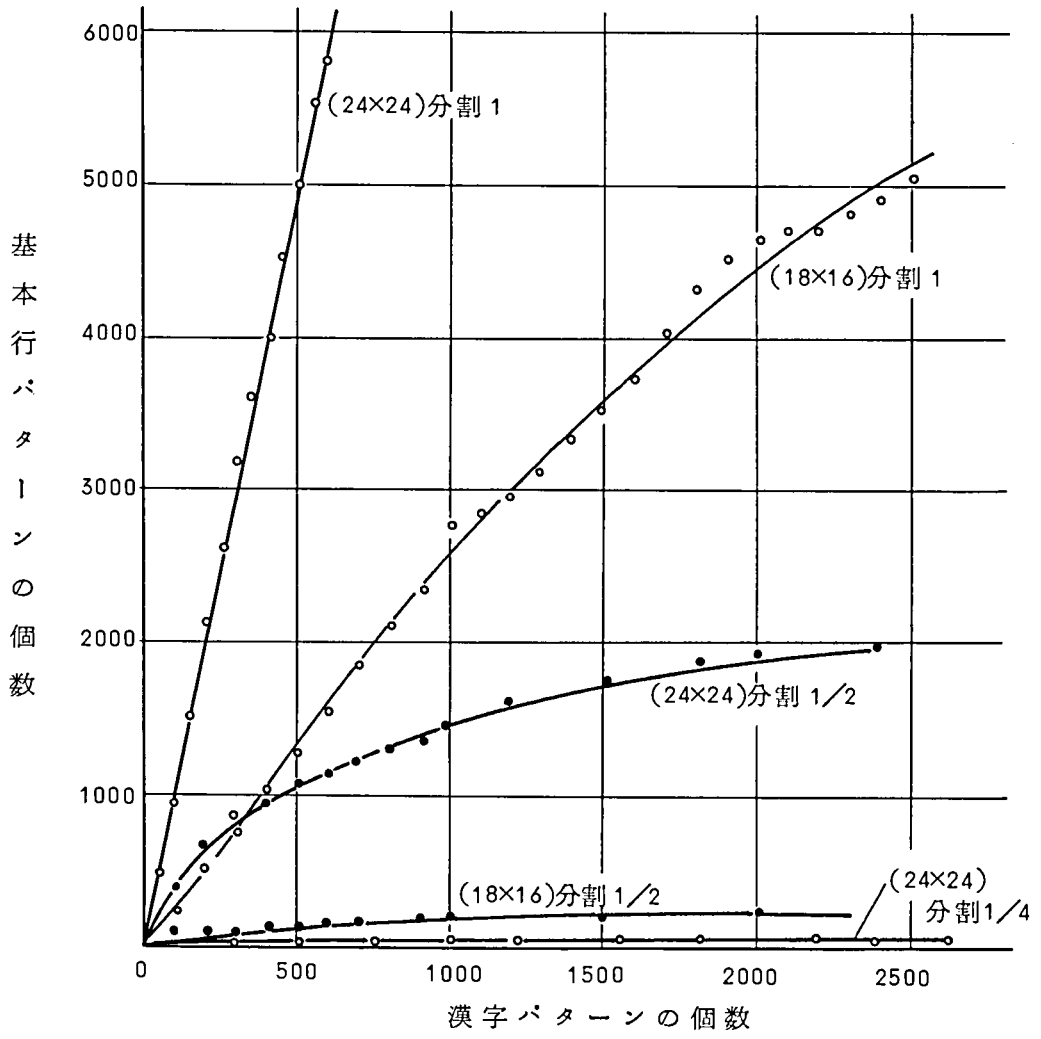


図2.2　漢字パターンの集合 $\{A^j\}$ の大きさと基本行パターンの集合 $\{B_k\}$ の大きさの関係

１行パターンについてみると，（２４×２４）の漢字パターンに対しては約５００字種で基本行パターンの個数が5,０００個となり，１字当り平均１０個の割で基本行パターンの個数は増加する。また，（１８×１６）の漢字パターンでは約1,０００字種，2,５００個の基本行パターン数までは１字当り平均2.5個の基本行パターンの割合で字種数に比例して基本行パターンの個数は増加しているが，それ以上の字種の増加に対しては次第に頭打ちとなる傾向が明らかとなる。しかし，その飽和状態への移

行は緩慢であり，行パターンの長さがnではあまり大きな圧縮効率は期待できないことが分かる。

次に1/2行パターンについてみると，（２４×２４）と（１８×１６）の漢字パターンに対してそれぞれ，基本行パターンの個数は約2,０００個と２５０個で飽和する。

さらに，1/4行パターンでは（２４×２４）と（１８×１６）の漢字パターンのいずれにおいても，それぞれ出現可能な６４種類と１６種類の基本行パターンが全て出現することが分かる。

図2.3に図2.2と同じ漢字パターンの集合に対して得られた基本行パターンの出現

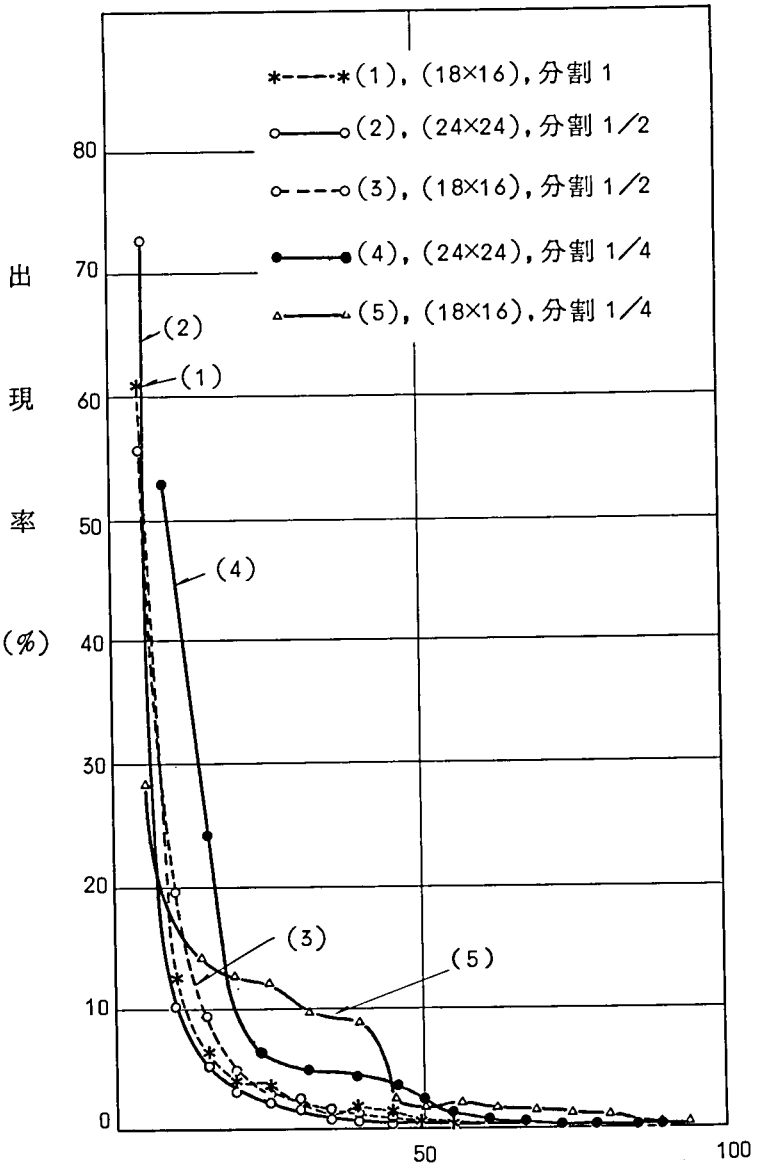


図2.3　基本行パターンの出現頻度分布

ひん度分布を示す。縦軸は各基本行パターンを用意することによって，漢字パターンの集合$\{A^j\}$に対応して必要となる行パターンをカバーできる割合である。

表2.1には基本行パターンの集合における特徴的な数値を示している。

表2.1　基本行パターンの集合における特徴的な数値

漢字パターン	分割	文字数	基本行パターン数	基本行パターン率	平均出現頻度	最高出現頻度	出現頻度1の基本行パターン数
(18×16)	1	1,734	4,080	6.4	7.7	1,509	1,723
	1/2	1,763	240	93.8	264.5	6,999	13
	1/4	1,700	16	100	7,650	34,230	0
(24×24)	1	1,000	9,506	0.05	9.5	721	7,176
	1/2	2,366	1,993	48.7	48	10,199	418
	1/4	2,400	64	100	3,600	60,602	0
(32×32)	1/2	3,008	3,961	6.0	48.6	49,534	1,019
	1/4	3,008	233	91.0	1,652	158,578	13
	1/8	3,008	16	100	48,128	461,694	0

図2.3と表2.1から分かるように，1行パターンの出現ひん度分布曲線は低出現ひん度の方へ長く裾をひいており，(18×16)の漢字パターンでも出現ひん度1の基本行パターンが全基本行パターンの4割以上にも達している。

なお，表2.1で基本行パターン率とは次のように定義している。

$$\text{基本行パターン率} = \frac{\text{基本行パターンの個数}}{2^n} \times 100 \qquad (2.7)$$

2.2.3　行パターンの最適分割

式(2.5)で定義した漢字パターンのデータ圧縮比を高めるためには式(2.4)で与えられるデータ量Qを小さくすることが必要である。

総データ量Qは基本行パターン自体を表現するのに要するデータ量Q_1と式(2.3)に示すように，各漢字パターン毎に基本行パターンの組み合せ方を示す符号データ量Q_2の和で与えられるから，この両者の和として最少化を行なうことが必要である。

しかし，従来報告されている部分パターンへの分割と符号化によるデータ圧縮法の

考察においては部分パターンの大きさに対する考察はなされておらず，比較的小さな，少数の画素を含む部分パターンを適宜設定することによりデータ量Q_1を無視し，部分パターンの大きさによるデータ量Q_2の最少化を議論の対象としてきた。[4-2][14][22][54]〜[57]

しかし，Q_1とQ_2は全く独立な量ではなく，部分パターンの大きさ，すなわち，ここでは行パターンの長さによって互いに関係しているため，$Q = Q_1 + Q_2$の最少値はQ_1を固定した場合と，そうでない場合は当然異なったものとなる。すなわち，後者の場合が一般により小さな値となる。

そこで，以下，データ量Qを最小とする行パターンの分割について考察する。

(1) 行パターンの分割と情報量

漢字パターンを複数個の部分パターンに分割した時，1つの部分パターンの出現確率は周囲の部分パターンの影響を受け，部分パターンの間には互いに相関がある。

隣接した2つの部分パターンD_1，D_2と，それらの2つの部分パターンを結合した部分パターンD_3を考える。この時，部分パターンD_3がもつエントロピー$H(D_3)$は一般に次式で与えられる。[14]

$$H(D_3) = H(D_1) + H(D_2) - I(D_1, D_2) \qquad (2.8)$$

ここで，$H(D_1)$, $H(D_2)$はそれぞれ部分パターンD_1，D_2のエントロピーであり，$I(D_1, D_2)$は2つの部分パターンD_1とD_2の間の相関の強さを表わす相互情報量である。

2つの部分パターンD_1とD_2が確立的に等価であれば$H(D_1) = H(D_2)$であり，式(2.8)は

$$H(D_3) = 2H(D_1) - I(D_1, D_2) \qquad (2.9)$$

となる。

式(2.8)は互いに相関をもつ部分パターンがある場合には，個々に各部分パターンを符号化するよりも，相関をもつ部分パターンを結合した部分パターンを作り符号化を行なった方が符号データ量は少なくてすむことを意味している。

そこで，(18×16)のゴシック体漢字パターンと(24×24)，(32×32)の明朝体漢字パターンを対象に行パターンの分割と行パターンがもつ情報量の関係について調べた。

表2.2にその結果を示す。

表 2.2　行パターン間の相関

漢字パターン / 分割 / 行パターンの長さ / 情報量（ビット）	（18×16）ゴシック体			（24×24）明朝体			（32×32）明朝体		
	1/4	1/2	1	1/4	1/2	1	1/8	1/4	1/2
	4	8	16	6	12	24	4	8	16
エントロピー	3.16	5.83	10.70	4.21	7.76		2.25	4.16	7.49
相互情報量	0.49	0.96		0.66			0.34	0.83	

同表で相互情報量は式（2.9）から

$$I(D_1, D_2) = 2H(D_1) - H(D_3) \qquad (2.10)$$

で算出したものであり，各部分パターンD_i（$i=1, 3$）がもつエントロピー$H(D_i)$は

$$H(D_i) = -\sum_j P_{ij} \log_2 P_{ij} \qquad (2.11)$$

で求めたものである。ここでP_{ij}は部分パターンの集合D_iの要素D_{ij}の生起確率である。

さらに，表2.2に示したエントロピーのデータを用いて式（2.6）に示した，符号データのみを対象とした理論的なデータ圧縮率η_2の下限を求めた結果を表2.3に示す。

表2.3　符号データ量Q_2に対する理論的圧縮率η_2

漢字パターン / 分割 / 圧縮率	（18×16）ゴシック体			（24×24）明朝体			（32×32）明朝体		
	1/4	1/2	1	1/4	1/2	1	1/8	1/4	1/2
η_2	0.790	0.728	0.669	0.701	0.647		0.563	0.520	0.468

表2.2から1/2行パターン，1/4行パターンのそれぞれの配列の間も明らかに相関を有していること，その結果，表2.3に示すように実際の漢字パターンにおいても，符号データ量の面からいう限り，部分パターンは大きい程よく，1行パターンが有利

となる。

(2) 基本行パターンの集合 $\{B_k\}$ がもつ冗長さ

行パターンの長さの増加に伴い，理論的に出現可能な行パターンの数は2のべき乗で増加する。それに対応して実際の漢字パターン内に現われる基本行パターンの数も急激に増加することは図2.2に示されている。

しかし，基本行パターンとして出現が許されるパターンの種類は，漢字パターンを構成する行パターンという潜在的な条件から限られるため，両者の増加の割合いにはおのずと差がある。したがって行パターンの長さが長くなれば，理論的に出現可能な行パターンの個数内に占める基本行パターンの個数の割合い，すなわち，式(2.7)で定義した基本行パターン率は小さくなる。

このことは，行パターンを構成する各画素が担う情報量が小さくなることを意味する。そこで，以下，行パターンを構成する各画素が担う情報量の観点から行パターンの長さについて考察する。

行パターンは白画素と黒画素の組み合せである。同一種類の画素が1個以上連続している場合，その連なりをラン，連なった画素数をランレングスとよぶ。さらに，白画素のランを白ラン，黒画素のランを黒ランとよぶ。

長さを与えられた行パターンを考えるとき，行パターンを特徴づけるパラメータとしては白，黒の画素数比，黒(又は白)ランの個数と位置，ランレングスの分布がある。

これらのパラメータのうち，白，黒画素数比，黒ランの位置，ランレングス分布は漢字パターンの構成画素数，字体，字肉の太さの異なる漢字パターンに対して異なった値をとることになる。一方，漢字パターンの構成画素数，字体，字肉の太さなどの字形の異なる漢字パターンにおいても漢字を構成する字画数は不変である。行パターンとしてみたときこの性質は黒ラン数(又は白ラン数)の分布の不変性に対応する。すなわち，漢字パターンの構成画素数，字体，字肉の太さが異なっても，同一字種の同一位置を占める行パターンであれば行パターン内のラン数は不変である。

表2.4は(18×16)のゴシック体漢字パターンの1,763字に対して調べた基本行パターン内の黒ラン数の分布である。この分布の形状を図2.4に実線で示している。

表2.4　基本行パターン内における黒ラン数の分布

分　割	黒　ラ　ン　数　(%)								
	0	1	2	3	4	5	6	7	8
1	0.03	3.48	22.01	33.00	24.96	12.48	3.48	0.57	0.03
1／2	0.41	15.00	51.67	29.17	3.75				
1／4	6.25	62.50	31.25						

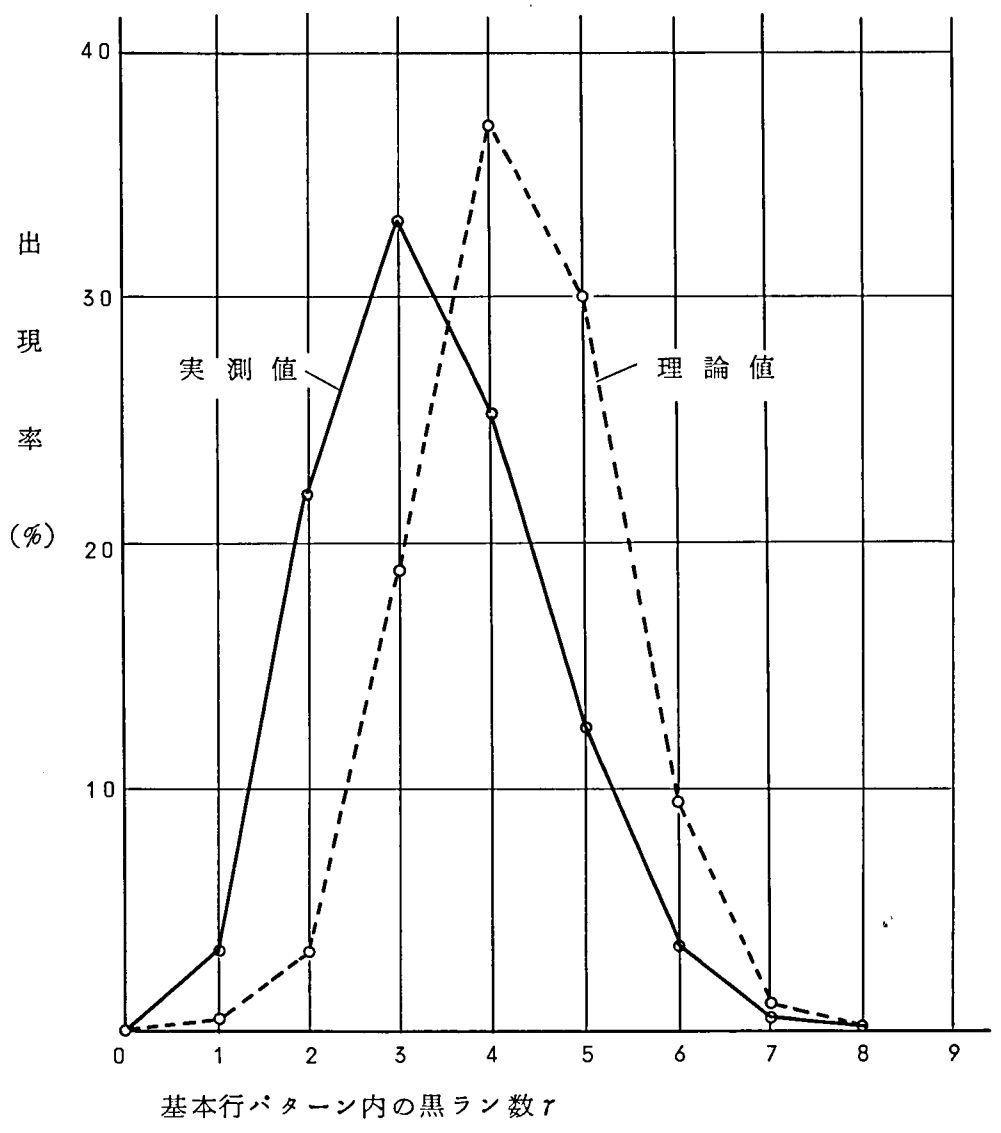


図2.4　基本行パターン内の黒ランの個数分布
（試料：18×16ゴシック体漢字パターン）

一方，長さnの行パターン内に，位置，長さともに異なるγ個の黒ランを配置する方法の数$N(n, \gamma)$は

$$N(n, \gamma) = \frac{1}{(2\gamma)!}(n-2\cdot\overline{\gamma-1})\cdots\cdots n\cdot(n+1)$$

$$\gamma \geqq 1 \qquad (2.12)$$

で与えられる。（付録2.1）

$n=16$の場合について式（2.12）に従った黒ラン数の分布を図2.4に破線で示している。

図2.4に破線で示す式（2.12）で与えられる黒ランの分布は全く制約の無い状態で黒ランを配置した場合に生じる分布であり，漢字パターンにおいても制約がなければ，黒ランの分布は当然この分布にしたがうはずである。このことから，図2.4において実測値分布の理論値分布からのずれが漢字パターンの規則性の程度に対応すると考えられる。

行パターンの長さが24，32と長くなれば理論値の分布は，ラン数γの大きい方へとさらにずれ，ピークはそれぞれ$\gamma=6$，$\gamma=8$でとるようになる。

ここで次の関数を定義する。

$$R(n, \gamma) = \frac{N(n, \gamma)}{2^n} \qquad (2.13)$$

関数$R(n, \gamma)$は，長さnの行パターン内にγ個の黒ランを含む行パターンの個数が長さnの理論的に使用可能な行パターンの全個数内に占める比率を与えている。

例えば，$R(24, 3)$は3個の黒ランを含むパターンの全てを長さ24の行パターンを用いて作った場合の行パターンの構成画素の使用効率の理論値を示している。

図2.5は関数$R(n, \gamma)$をγをパラメータとして図示したものである。

いま，黒ランを3個までしか含まないパターンを長さ24の行パターンを用いて作成する場合の行パターンの構成画素の使用効率Cは高々

$$C = R(24, 1) + R(24, 2) + R(24, 3)$$

$$\fallingdotseq 1.13\times 10^{-2} \qquad (2.14)$$

にすぎないことになる。しかも，これはγの値が1，2，3のとき式（2.12）で与えられる個数のすべての行パターンが使用し尽くされたときに達する使用効率の上限

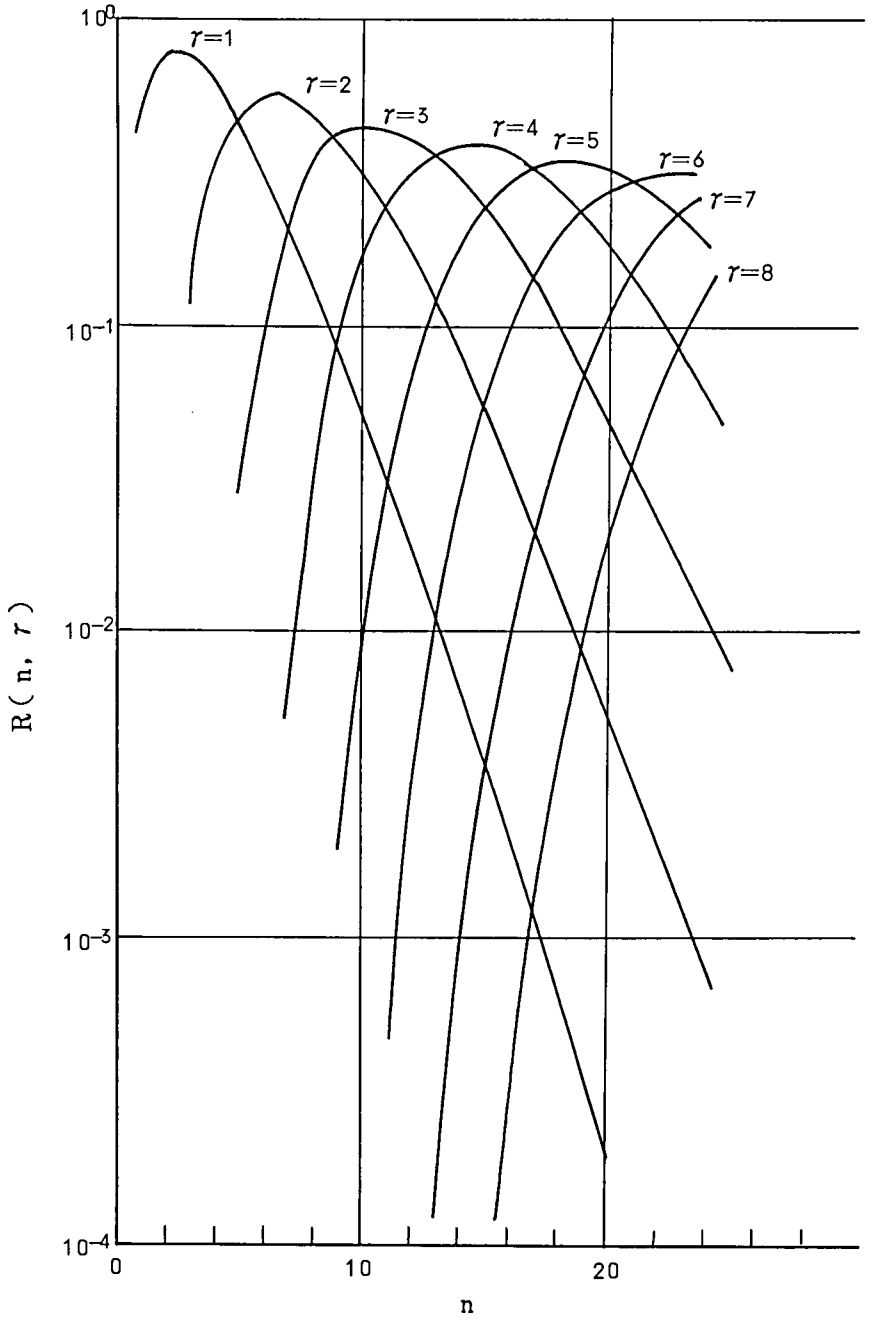


図 2.5　関　数　$R(n, r)$

である。

しかし，実際の漢字パターンにおいては表 2.4 に示すように，$N(n, r)$ で与えられる個数の行パターンを r の小さな方から順次完全に使用し尽していくことはない。

したがって，行パターンの構成画素の使用効率は次式で与えられることになる。

$$C = \sum_{\gamma=1}^{\gamma_{max}} \frac{S(\gamma)}{N(n,\gamma)} \cdot R(n,\gamma) \qquad (2.15)$$

ここで，$S(\gamma)$は漢字パターンにおいて現われた，黒ランをγ個含む行パターンの個数，γ_{max}は1個の行パターン内に含まれた最大の黒ラン数である。

式（2.15）は式（2.13）を用いて

$$C = \sum_{\gamma=1}^{\gamma_{max}} \frac{S(\gamma)}{2^n}$$

$$= \frac{1}{2^n} \sum_{\gamma=1}^{\gamma_{max}} S(\gamma) \qquad (2.16)$$

となる。

すなわち，行パターンの構成画素の使用効率の観点からは，「行パターンの長さnは，行パターンの長さをnとすることによって出現する基本行パターンの数が2^n個か，又は，それに近い個数となるように選ぶ必要がある。」ことを意味している。

表2.5に表2.1に示した（18×16）の漢字パターンについて求めた行パターンの構成画素の使用効率Cを示す。

表2.5　行パターン構成画素の使用効率
（18×16）漢字パターン

分　　割	1	1／2	1／4
効　率　C　(%)	6.4	94	100

(3)　行パターンの最適分割数

前記(1)で述べたように，符号データ量Q_2を減らすためには行パターンの長さは長い方が有利であるが，基本行パターンの集合$\{B_k\}$のデータ量Q_1内に含まれる冗長さの点からすれば，許容される行パターンの長さには限界があることを前項で述べた。

これらの2つの条件より，総データ量Qを最少とする行パターンの最適分割数を決定することができる。

上述の2つの条件は，等しい重みをもつものではなく，行パターンの長さに依存した行パターン間の相関の程度，基本行パターンの集合$\{B_k\}$の大きさにより異なる。

図2.6に（18×16），（24×24），（32×32）の漢字パターンについ

て，行パターンの分割に伴うデータ圧縮率ηの変化を図示する。η_1は表2.1に示す基本行パターン数から求め，η_2は表2.3に示す値を用いている。

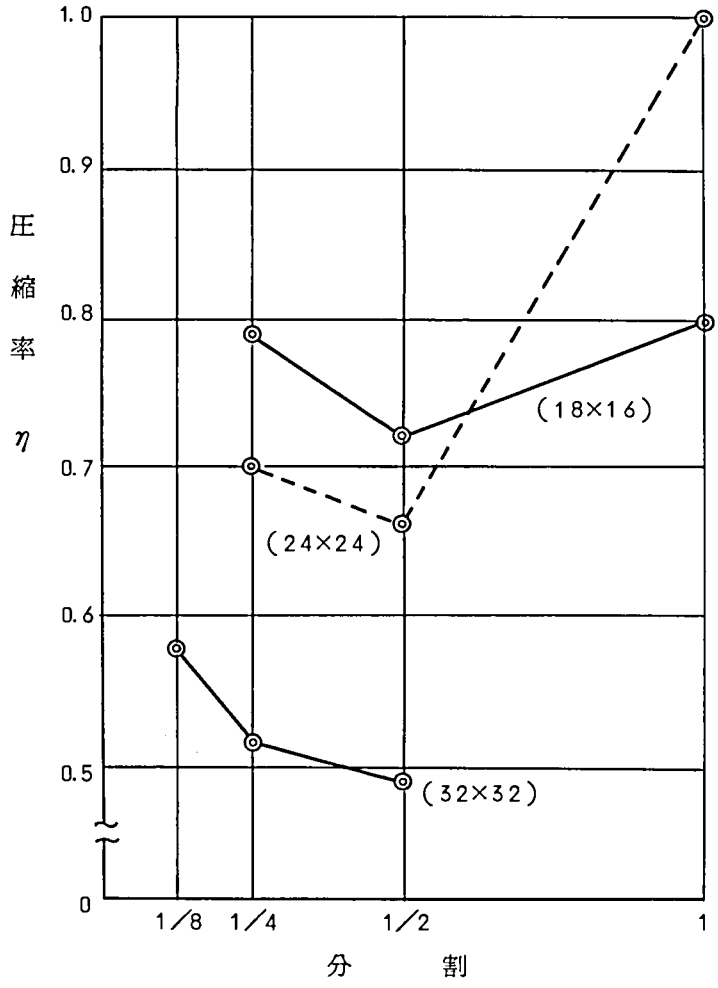


図2.6　行パターンの分割とデータ圧縮率

なお，（２４×２４）の漢字パターンの１行パターンについては，1/2行パターン間の相互情報量を（１８×１６）の漢字パターンの場合から類推して１ビットと仮定している。

以上の検討結果，本検討で用いた文字数の範囲では，（１８×１６），（24×24），（３２×３２）の全ての漢字パターンに対して行パターンの分割としては1/2行パターンが適当であり，その時，それぞれデータ量が７７％，６６％，４９％に圧縮されることがわかる。

分割数としては（１８×１６），（２４×２４），（３２×３２）の全ての漢字パターンに対して２が最大の圧縮効果を与えるが，図2.6に示すデータ圧縮率ηの変化

を見ると次のことがいえる。

「行パターンの長さは，基本行パターン率が１となる範囲では出来るだけ長く，基本行パターン率が１より小さくなる範囲では出来るだけ短かく設定する方が，より高いデータ圧縮効果が得られる。」

すなわち，行パターン合成符号化法では，漢字パターンの構成画素数に応じて変化する最適な行パターンの長さが存在する。

2.2.4　符　号　化

前項で，行パターンの最適分割数を求め，その時得られるデータ圧縮率を，理論的な限界としてエントロピーを用いて算出した。

ここでは，基本行パターン B_k に与える符号 C_k の具体的な構成について述べる。

出現頻度に差がある事象の符号化としてはハフマンの符号化法が高い符号化効率を与えることが知られているが，ハフマンの符号は完全な不等長符号となり，復号過程で復号器の構成が複雑となり，数千に及ぶ事象を対象とする場合には現実的でない。

そこで図2.7に示す符号構成の可変長符号化を行なった。すなわち，最短符号長を l ビットとし，符号化すべき基本行パターンの出現頻度の順位に応じて，その整数倍の長さを与えるものである。最短符号長の L 倍の符号長を有する符号の構成は，符号の先頭より（$L-1$）個の１を並べ，L 番目のビットに０を置く。したがって $2^{(l-1)\cdot L}$ 個の基本行パターンの符号化が可能となる。

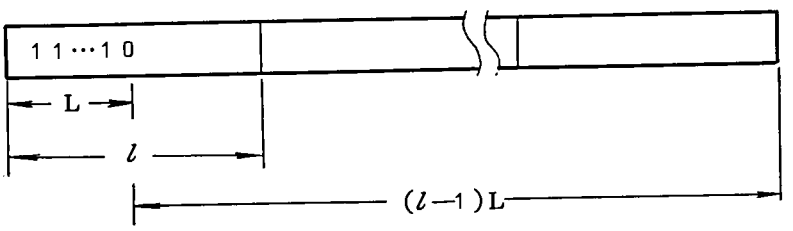


図2.7　可変長符号構成

表2.6に最短符号長 l を変えたときの圧縮率 η_2 の変化を示している。

符号化効率を表2.3に示した1/2行パターンに対する圧縮率 η_2 と表2.6に示した圧縮率 η_2 との比として求めると，$l=3$ のとき，（１８×１６）の漢字パターンに対して0.942，（２４×２４）の漢字パターンに対しては0.926が得られる。

表2.6　最短符号長 ℓ と圧縮率 η_2

字　体	（18行×16列）				（24行×24列）			
ℓ（ビット）	3	4	5	6	3	4	5	6
η_2	0.773	0.813	0.854	0.890	0.699	0.701	0.715	0.743

2.2.5　2次元相関の導入

以上，部分パターンを一次元の行パターンとすることにより，漢字パターンの構成画素間の横方向への相関は利用することができた。

しかし，漢字パターンは2次元パターンであり，データ圧縮効率を高める上で縦方向への相関を無視することは極めて大きな損失となる。すなわち，アルファベット，数字，ひら仮名，さらに一般の図形に比べて漢字パターンは直線分を多数含んでおり，かつ，それらの直線線分の向きが縦と横方向に強く偏っていること，また，いわゆる偏や旁などの2個以上の部首から合成された漢字パターンが多いこと，といった特徴から，1個の漢字パターン内で縦方向へ同一の1/2行パターンが複数個連続する場合が多く現われる。

そこで，多数のラスタ走査形出力装置で1台の漢字パターン発生器を共有するという立場を離れて，2次元相関を利用した場合の行パターン合成符号化法のデータ圧縮率を次に考察する。

表2.7は（24×24）の漢字パターンについて，縦方向へ隣接する1/2行パターンに関して，白，黒異なった状態の画素数の分布を求めた結果である。

全く同一の1/2行パターンが並ぶ場合が約25%，3画素以内の差異しかない場合が約67%にも達することが分かる。

表2.7　隣接する1/2行パターン間の変化画素数分布

変化画素数	0	1	2	3	4	5
分　布　(%)	25.36	14.87	17.63	9.45	6.93	5.42

6	7	8	9	10	11	12
5.47	4.76	4.04	2.60	2.03	0.79	0.64

この結果，縦方向へ隣接する1/2行パターン間の差を処理する方法も有効になると考えられるが，漢字パターンの発生はあくまで1/2行パターンを単位として完結させることを考え，符号の連続として処理できる同一の1/2行パターンが縦方向に連続する部分のみを考慮することとした。

図2.8は（２４×２４）の漢字パターンについて1/2行パターンの縦方向への連続数の分布を調べた結果を示したものである。

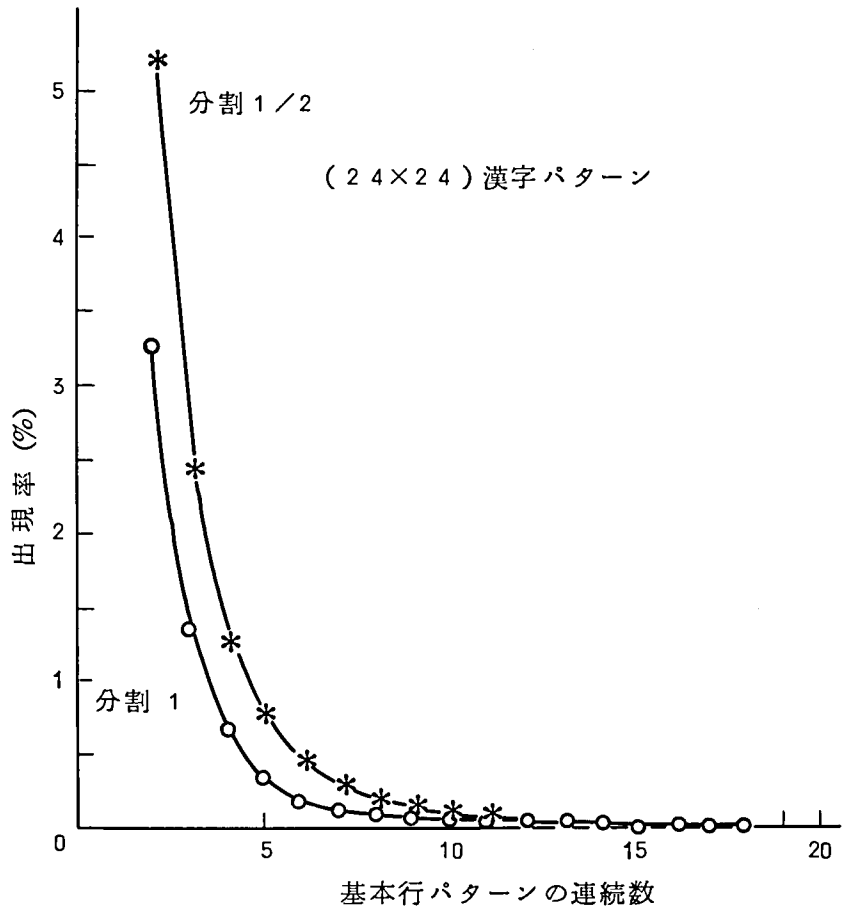


図2.8　行パターンの縦方向への連続数分布

この縦方向への1/2行パターンの連続部分を連続数を指定する符号と連続する1/2行パターンの符号の組で行なうこととした。

具体的な符号構成の例として，連続数の指定符号を図2.7に示した最短符号長で実現するため，最短符号長を $l = 5$ とし，図2.9の構成とした。このとき，1/2 行パターンの連続性を考慮した効果として，（２４×２４）の漢字パターンに対して７０ビット／字のデータ量が削減され，約１２.２％のデータ圧縮率の改善が得られた。

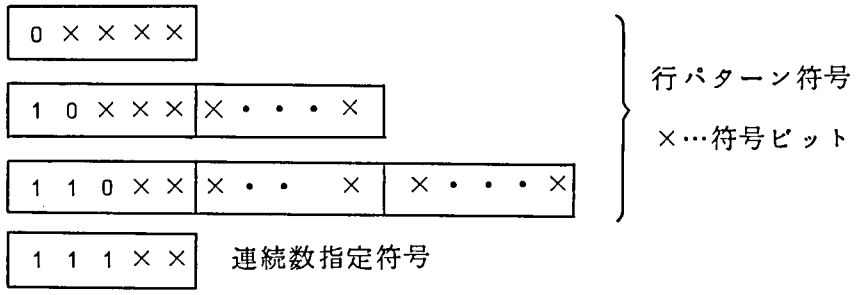


図 2.9　連続数指定符号も考慮した符号構成

この結果，最短符号長 $\ell = 5$ のとき，符号データ量のみの圧縮率として $\eta_2 = 0.593$，基本行パターンの集合 $\{B_k\}$ がもつデータ量に対して $\eta_1 = 0.018$ となり総合の圧縮率として $\eta = 0.611$ が得られ，データ量を約４０％削減することができた。

（１８×１６）の漢字パターンについても同様に，最短符号長を $\ell = 5$ としたとき，1/2 行パターンの連続を考慮することにより３６ビット／字，約１２.５％のデータ圧縮率の改善が得られ，総合の圧縮率 $\eta = 0.73$ となり，約３０％のデータ量が削減される。

2.2.6　復号装置の構成

漢字パターンデータ圧縮処理には発生速度の低下と復号処理部のハードウェアの増加が伴なう。従ってデータ圧縮法の評価では発生速度の低下とハードウェア量の増加を評価することが必要である。

そこで，本項では行パターン合成符号化法の発生速度とハードウェア構成について述べる。

(1)　漢字パターンの符号化データ

図２.１０に示す漢字パターンを例にとって，漢字パターンの符号化データについて述べる。

図 2.1 の(a)に示す（１８×１６）の漢字パターン「禅」は図２.１０に示すように３６個の 1/2 行パターンへと分解される。

各 1/2 行パターンはそれぞれの出現頻度に応じて図 2.9 に示した構成の符号が割り当てられる。ただし，図２.１０では 1/2 行パターンに与えられた符号をアルファベットで代表し，Ａ～Ｉで示している。その結果，1/2 行パターンの縦方向への連続性を

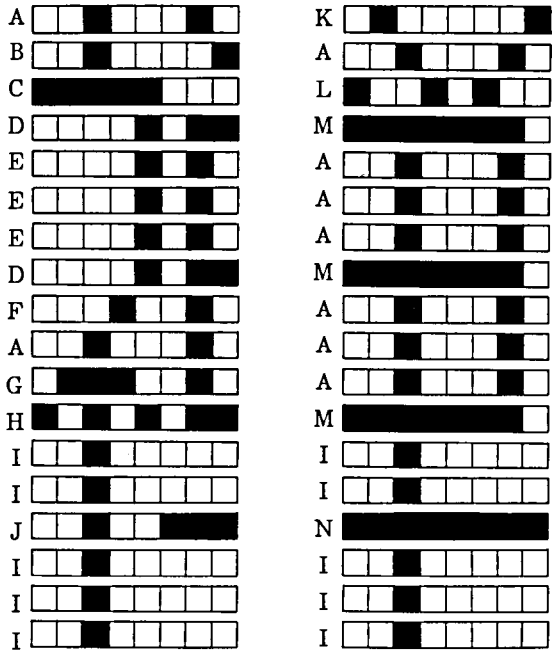


図 2.10　1/2 行パターンへ分割された漢字パターン

考慮しない場合の漢字パターンの符号化データとして図 2.11 に示す符号列が得られる。図 2.11 で符号列の上側，下側にかっこで結んだ同一符号の組が図 2.10 の漢字パターンで見られる 1/2 行パターンの縦方向への連続部分に対応している。

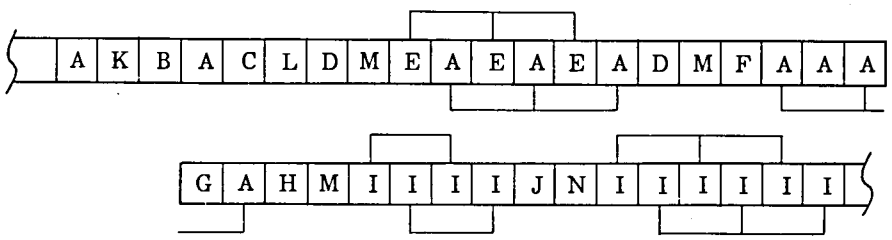


図 2.11　縦方向への連続性を考慮しない時の「 禅 」の符号列

これらの縦方向へ 1/2 行パターンが連続する部分は，図 2.9 に示した連続数指定符号を用いて符号化する。

その結果，最終的な漢字パターンの符号化データは図 2.12 に示すようになり，この符号化データが漢字パターン発生器内に格納されることになる。なお，図 2.12 では＊は連続数指定符号を，その後に続く数字は連続数を示している。

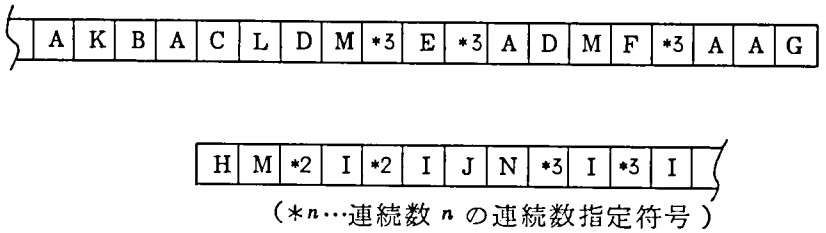


図 2.1 2　縦方向への連続性を考慮した時の「 禅 」の符号列

(2)　漢字パターン発生器の構成と動作[58]

行パターン合成符号化方式を適用した漢字パターン発生器の構成ブロック図を図 2.1 3 に示す。主要な構成は符号データメモリ（以下，ＣＭと記す），行パターンメモリ（ＲＰＭ），ＣＭアドレスレジスタ，ＲＰＭアドレスレジスタ，連続数レジスタ，アドレス切換部，それに全体を制御する制御部からなっている。

ＣＭには図 2.1 2 に示した漢字パターンの符号データが，ＲＰＭには基本 1/2 行パターンが画素パターンとして格納されている。

ＲＰＭ内での 1/2 行パターンの格納順序は出現頻度の順に従っている。すなわち，図 2.9 に示した 1/2 行パターンの符号がそのまま格納されているＲＰＭのアドレスとなるように構成する。

連続数レジスタには連続数指定符号の連続数が格納される。

つぎに動作の概要について述べる。

漢字パターンの発生は，第 1 行目から左，右，左，右の順で順次 1/2 行パターンを発生させるのが基本である。したがって，ただ，縦方向へ同一の 1/2 行パターンが連続する場合の制御が発生器に必要な判断制御機能となる。

すなわち，縦方向へ同一の 1/2 行パターンが連続する部分が現われると，それまで左，右の順で並んでいた行パターン符号の順が変わり，各 1/2 行パターン符号が漢字パターンの左右のいずれに対応するのかの判断を行なうことが必要となる。

しかし，この判断は連続数レジスタの内容をみて容易に行なわれ得る。図 2.1 3 に示すように，ＲＰＭアドレスレジスタと連続数レジスタを対として左，右 2 組用意しておき連続数レジスタの内容は，対をなすＲＰＭアドレスレジスタの内容が以後，引続いて使用される回数を示すように構成しておくことにより，ＣＭから読み出された 1/2 行パターン符号は連続数レジスタの内容が 0 になっているＲＰＭアドレスレジス

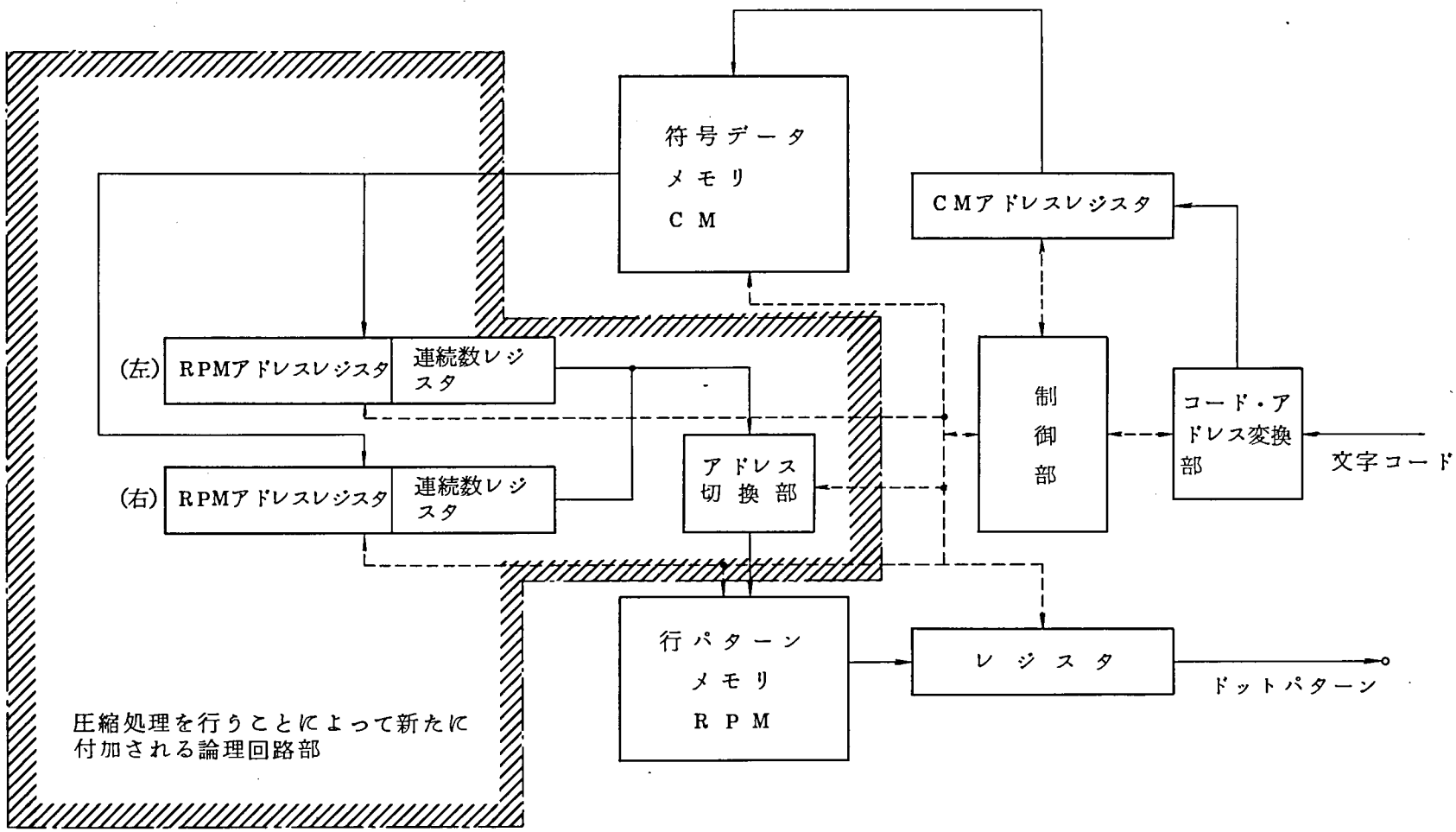


図2.13　行パターン合成符号化法を適用した漢字パターン発生器の構成ブロック図

タに格納すれば良いことになる。左，右の連続数レジスタの内容がともに0であれば左側のＲＰＭアドレスレジスタに格納することにする。

したがって動作はつぎのようになる。入力として受取った漢字コードはコードアドレス変換部で指定された漢字パターンの符号データを格納したＣＭの先頭アドレスへと変換され，ＣＭアドレスレジスタに格納される。

その後，

① 漢字パターンの発生開始時点に左，右の連続数レジスタがクリアされる。

② その後，左右の順序で連続数レジスタの内容を調べ，左連続数レジスタの内容が0であればＣＭアドレスレジスタ内のアドレスを用いてＣＭの内容を読み出す。制御部では読み出された内容を1/2行パターン符号と連続数指定符号に分離し，それぞれ左側のＲＰＭアドレスレシスタと連続数レジスタにセットする。

連続数指定符号がない場合は連続数は1とする。

この後，ＣＭアドレスレジスタの内容を次に読み出すべき符号のアドレスを示すように更新する。このアドレスの更新は読み出された1/2行パターン符号の符号長と連続数指定符号の有無を判定して行なわれる。

③ 右連続数レジスタの内容が0であれば，ＣＭアドレスレジスタ内のアドレスを用いてＣＭの内容を読み出し，②と同様に，1/2行パターン符号と連続数指定符号をそれぞれ，右側のＲＰＭアドレスレジスタと連続数レジスタにセットする。その後，ＣＭアドレスレジスタの内容を更新する。

④ 左，右両方の連続数レジスタが1回ずつセンスされた後，ＲＰＭアドレスレジスタ内のアドレスを用いて左，右の順序でＲＰＭから1/2行パターンが読み出され，出力装置へ送出される。また，同時に，左，右の連続数レジスタの内容が1だけ減ぜられる。

以上が漢字パターン発生の基本サイクルであり，②，③，④のステップを繰り返す。

②，③のステップで連続数レジスタの内容が0でなければ，そのまま次のステップへ進む。したがって，左，右の連続数レジスタの内容が共に0でない場合には，ＣＭからの符号の読み出しが無い基本サイクルとなる。

(3) 漢字パターンの発生速度とハード量

漢字パターンの発生速度を決めるおもな要因は，ＣＭあるいはＲＰＭを実現するメモリの読み出し時間と符号の識別と左，右のＲＰＭアドレスレジスタの切り換えをお

もな機能とする論理回路の動作時間である。

したがって，メモリの読み出し時間を τ_M，論理回路の動作時間を τ_L とすると，漢字パターンの発生速度 W（字／秒）は

$$W = \frac{1}{\tau_M + \tau_L} \qquad (2.17)$$

で与えられる。

メモリからの読み出し時間 τ_M は 1／2 行パターン 1 個を読み出すために必要なメモリへのアクセス回数を基に評価することができる。

1 回のアクセスによって符号データメモリＣＭから読み出すべきデータ量が最大となるのは出現頻度の小さな 1／2 行パターンが縦方向に連続する場合である。この時，図 2. 7 に示した符号構成に対して（ $L+1$ ）個の最短符号長のデータ量を読み出すことが必要となる。2. 2. 5 項で述べた最短符号長 $\ell = 5$，$L = 3$ の符号構成時には最大２０ビットとなる。

しかし，出現頻度の低い 1／2 行パターンが連続する場合が生じる確率は非常に小さく，多くの場合は最短符号長の符号を持つ 1／2 行パターンが出現することになる。

図 2. 3 に示した基本行パターンの出現頻度分布を用いて，$\ell = 5$ の場合の平均アクセス回数を求めると（２４×２４）の漢字パターンに対して約 1. 5 回となる。

したがって，1 個の 1／2 行パターンの発生に必要なメモリの読み出し回数は平均 2. 5 回となる。

一方，論理回路部の動作時間 τ_L は，1 個の 1／2 行パターンを発生する間に信号が通過するゲート段数で評価することができる。1／2 行パターン符号の検出部において３０段，行パターンメモリからの 1／2 行パターンの読み出しに２０段の計５０段が見積られる。

したがって，（ $m \times n$ ）の漢字パターンの発生速度 W（字／秒）は

$$W = \frac{1}{(2.5\tau_m + 50 \cdot \tau_g) \cdot 2m} \qquad (2.18)$$

τ_m ： メモリサイクルタイム（秒）

τ_g ： ゲートの伝播時間（秒）

となる。

今，$\tau_m = 1\ \mu$ 秒，$\tau_g = 10^n$ 秒，$m = 24$ として漢字パターンの発生速度を見積る

と

$$W = \frac{10^6}{(2.5 + 0.5) \times 48} \cong 6.9 \times 10^3$$

が得られる。

また，図2.13に示した構成で復号装置を試作した結果，符号データメモリ，行パターンメモリ，コードアドレス変換部を除いて約500ゲートのハードウェア規模で実現された。

2.2.7 ランレングス符号化法との比較

代表的な1次元符号化法としてランレングス符号化法がある。以下，ランレングス符号化法と行パターン合成符合化法と比較する。

(1) データ圧縮効率

図2.14に（24×24）と（32×32）の明朝体漢字パターンにおけるランレングス分布を示す。

このランレングス分布から，ランレングス符号化法の符号化効率は表2.8のようになる。

この結果，（24×24），（32×32）の漢字パターンに対しては，縦方向への行パターンの連続を考慮しない行パターン合成符号化法が

① 白黒ラン一括ランレングス符号化法に比較して圧縮率で11％

② 白黒ラン分離ランレングス符号化法に比較して，圧縮率で5～6％

高い圧縮効果をもつ。

なお，画素数の少ない（18×16）の漢字パターンに対してはランレングス符号化法でのデータ圧縮効果は期待できない。

試みに，「涙」，「励」，「卸」，「蚊」，「悔」，「敢」，「架」，「葉」，「忌」，「棋」の10文字に対して，白黒ラン一括ランレングス符号化法を適用した結果，データ圧縮率98％が得られた。

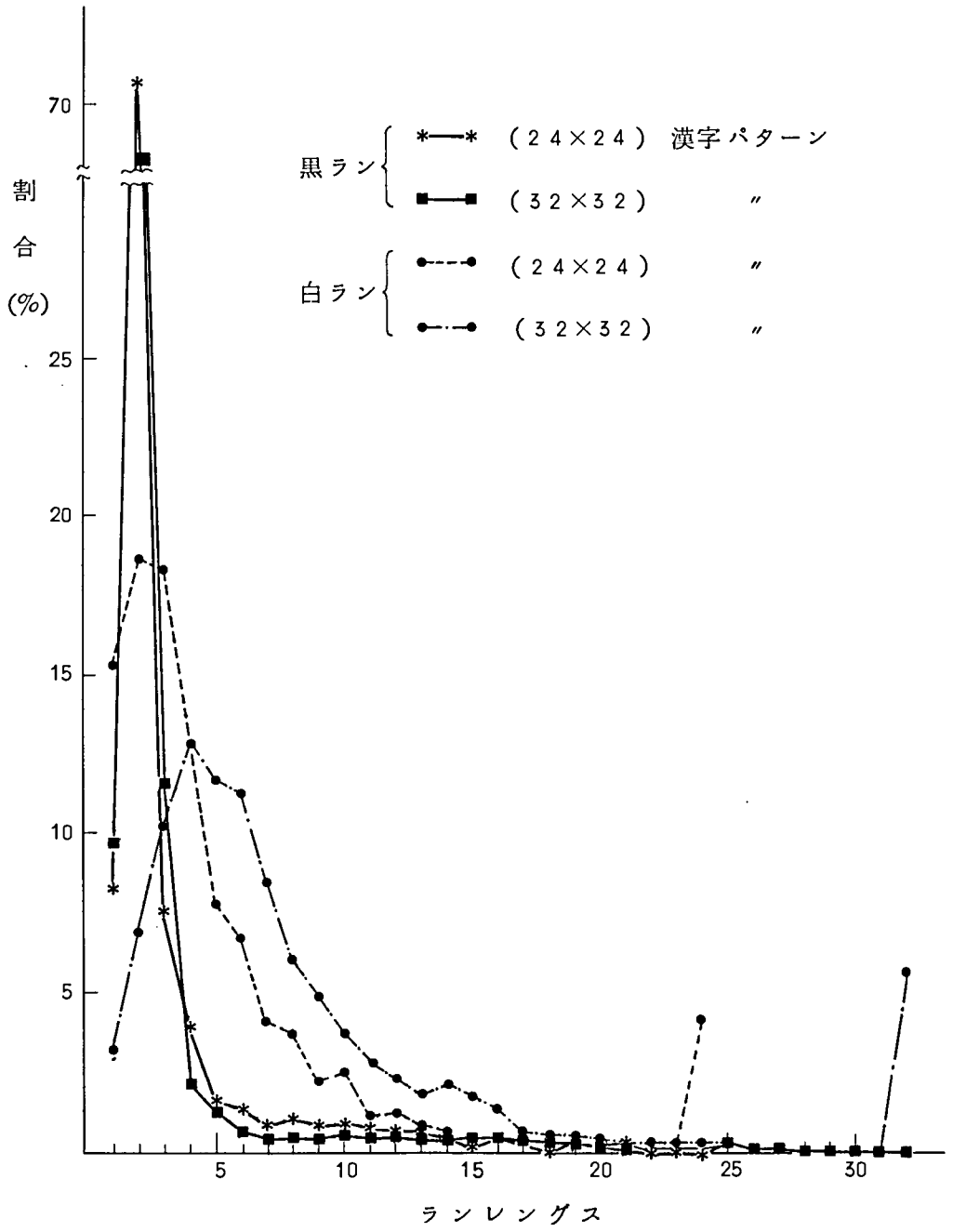


図 2.14　ランレングス分布

表2.8　ランレングス符号化法

項目	白，黒ラン一括		白，黒ラン分離			
			24×24		32×32	
	24×24	32×32	白ラン	黒ラン	白ラン	黒ラン
平均ランレングス	4.04	6.00	4.91	2.81	8.12	2.89
ラン当りのエントロピー（ビット）	2.93	3.38	3.35	1.80	3.95	1.82
1字当りのランの個数	142.8	170.8	83.2	59.7	101.3	69.5
*データ圧縮率 (%)	76.8	59.5	71.2		54.9	

＊）　各行パタンの先頭に，白，黒ランの識別用1ビットを含む。

(2)　パターン発生速度

図2.15に白黒ラン分離ランレングス符号化法の復号装置のブロック図を示す。行パターン1個の発生動作は次のようになる。

①　ランレングス符号メモリから行パターンの先頭ランのランレングス符号を読み出す。

②　先頭ランの白，黒をSW回路部で識別し，それぞれの符号・ランレングス変換テーブルを選択する。

③　符号・ランレングス変換テーブルを用いて符号をランレングス値へ変換し，カウンタとアダーへ送る。

④　白，黒ラン情報とランレングス値を用いてシフトレジスタ内にランを発生させる。

⑤　アダーでランレングス値を登算し，1行パターンの終了を検出する。

この結果，行パターンの発生速度を決める要因として，

①　ランレングス符号メモリの読み出し回数N_mとメモリのサイクルタイムτ_m

②　シフトレジスタ回路のシフトクロックf_S

③　符号・ランレングス変換テーブル用メモリのサイクルタイムτ_T

④　その他のSW回路部，カウンタ，アダー等の動作時間τ_L

がある。

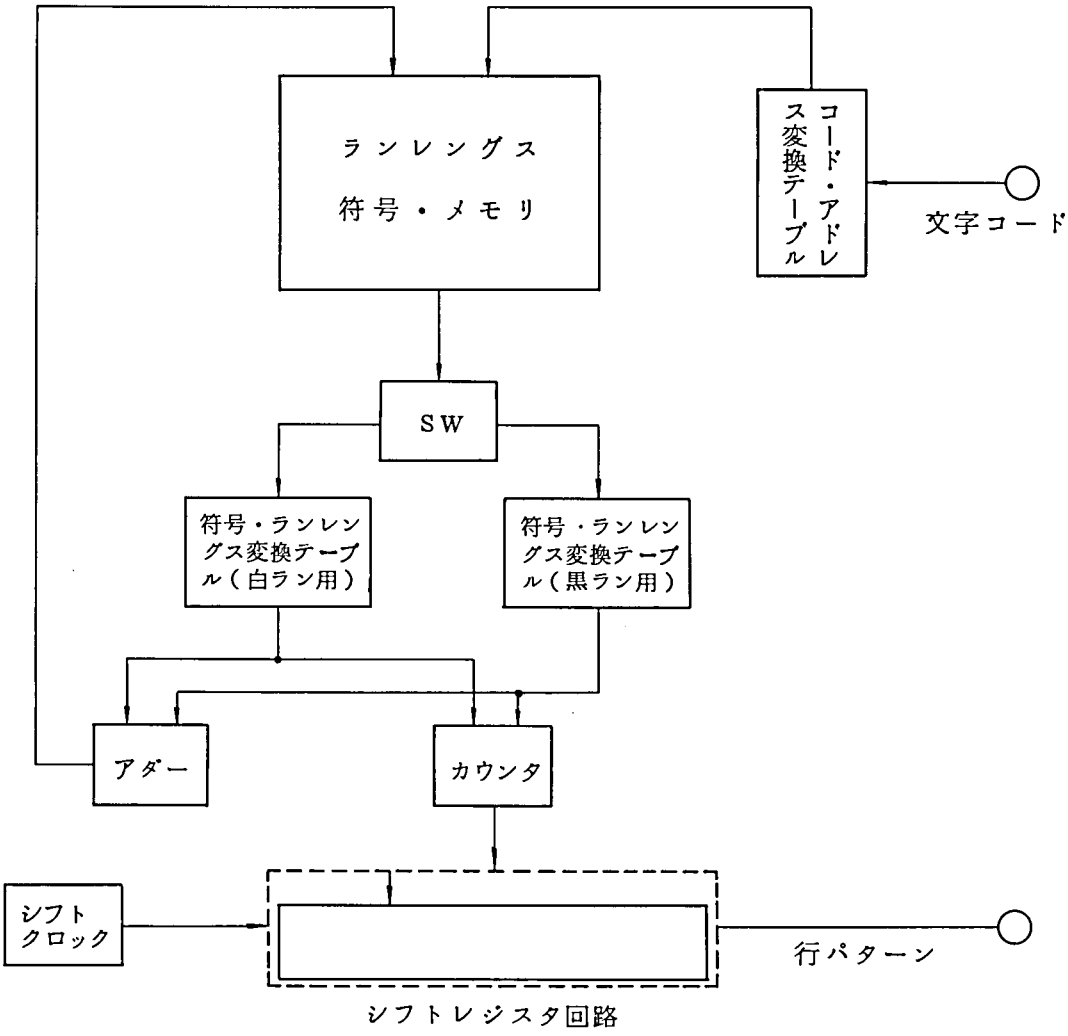


図 2.1 5　ランレングス符号復号装置構成

このうち，符号・ランレングス変換テーブルの読み出しとSW回路部等の動作は，シフトレジスタ内へのランの発生と並列に実行され得るが，ランレングス符号メモリの読み出しは，アダー回路での行パターンの終了判定結果を待って行なう必要があるため，シフトレジスタ回路の動作をシーケンシャルに行なう必要がある。

したがって行パターン1個の発生時間 τ_R の概略は

$$\tau_R \simeq N_m \cdot \tau_m + \frac{n}{f_S}$$

n ：行パターンの長さ（画素）

で見積ることができる。

$\tau_m = 1\ \mu$ 秒，$N_m = 7$，$n = 24$，$f_S = 10^6$ とすると

$$\tau_R = 3.1 \times 10^{-5} \text{ (秒)}$$

となる。

したがって，漢字パターンの発生速度W（字／秒）は

$$W = 1/3.1 \times 24 \times 10^{-5} \simeq 1.3 \times 10^{3}$$

となる。

この結果，ランレングス符号化法は行パターン合成符号化法に比較して文字パターンの発生速度は数分の１に低下することになる。

2.3 漢字パターンのサイズとデータ圧縮比

本格的な漢字情報処理システムの普及とともに，それらのシステムで必要とされる漢字パターンの種類は必然的に増加してゆくものと考えられる。

多種類の漢字パターンをシステム内に準備しようとする時，いかにメモリ価格が安価になってきたとはいえ，漢字パターンメモリのコストがシステム内で大きな割合いを占めることになる。

これらの問題の解決法として，漢字パターンのデータ圧縮技術や次章以降で述べる各種の漢字パターンの変換技術が研究されている。

本節では漢字パターンのサイズとデータ圧縮比の関係について考察する。

漢字パターンのデータ圧縮法については前節までに述べたように，すでに多くの研究が見られる。

データ圧縮手法の研究では高い圧縮効果を得ることが研究の目的であり，（１８×１６）（２４×２４）あるいは（３２×３２）とそれぞれのサイズの漢字パターンに対して，圧縮手法が研究されてきた。

しかし，複数種類の漢字パターンを含むシステムを構成するうえでは，各種の漢字パターン対応にデータ圧縮手法を変えることは望ましくなく，対象とする全ての種類の漢字パターンに同一の圧縮手法を適用したときに得られるデータ圧縮効果，すなわち，システムが用意すべき漢字パターンメモリ量の目安を得ることが必要となる。

そこで，本節では１次元相関を利用するランレングス形圧縮法と２次元相関を利用する境界線追跡形圧縮法の２つの符号化モデルを提案し，それらの符号化法について漢字パターンのサイズとデータ圧縮比の関係について定量的に考察する。

類似の研究としては，ファクシミリ信号の冗長度圧縮法の研究分野で，解像度と圧縮比に関する研究がある。(59)

しかし，ファクシミリ信号（電送すべき書画１枚分の標本化され２値化された信号を２値画像と見て，以下，ファクシミリ画像と呼ぶ）の統計的性質と漢字パターンの統計的性質は異なるため，ファクシミリ画像の議論をそのまま漢字パターンに適用することは出来ない。

すなわち，

① ファクシミリ画像の画素数は1,000×1,000～4,096×4,096程度あるのに対して，漢字パターンの画素数は１６×１６～１２８×１２８程度にすぎない。

② ファクシミリ画像は一般に任意の曲線から構成されているのに対して，漢字パターンでは構成線分に自由度が少ない。

すなわち，ファクシミリ画像の線分には線幅，方向共に自由度が多いが，漢字パターンでは幅，方向に制限が強い。

といった基本的な差異がある。

2.3.1 符号化モデル(27)

(1) 一次元符号化モデル（変形ランレングス法）

一次元の符号化法の代表はランレングス符号化法である。

一般の書画では１本の走査線上に出現するランの個数を予じめ知ることは出来ないため，走査線上のデータの区切りはランレングスを順次登算して，その値が前もって設定されている１走査線上の画素数と一致する点を検出することによって行なわれる。

一方，漢字パターンの符号化に際しては，予じめ走査線上，すなわち，行パターン内に含まれるランの個数を知ることができる。そこで，逆に，ランの個数を用いて行パターンを区切ることができる。

その結果，以下の点でデータ圧縮効率の改善が期待される。

① 全て白画素から成る行パターンに対応して現われる長い白ランが無くなり，より符号化に適した白ランの分布となる。

② 各行パターン内の最後の白ランを符号化する必要が無くなり，白ラン１個当りの符号量と，ラン個数指定に要する符号量との差の符号量が行パターン当り削減される。この削減される符号量は漢字パターンのサイズが大きくなる程大きくなる。

そこで，行パターンを，その中に含まれる黒ランの個数Nと，左から$2N$個のランのランレングスを用いて符号化する方法を変形ランレングス法と名付けて，一次元符号化モデルとする。

図2.１６は変形ランレングス法で実際にランレングス法が適用される範囲（以下，ランレングスエリアとよぶ）を示した図である。

ランレングスエリアが漢字パターンの全体に対して占める割合を符号化面積率とよぶ。

図2.１７に符号化の対象となる白ランのランレングス分布を，変形ランレングス法と従来のランレングス法とを対比して示している。

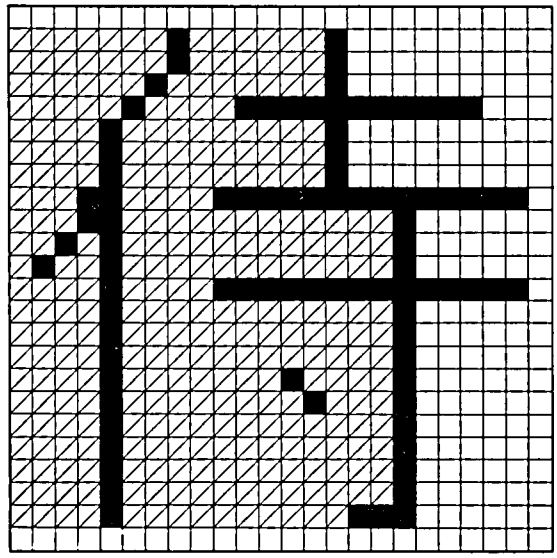


図2.１６　ランレングスエリア

また,データ圧縮率の評価に必要となる統計量を表2.９に示す。なお，用いた試料は（３２×３２）の明朝体漢字パターン８００字である。

表2.９から分かるように，白ランに関して平均ランレングス，エントロピー共に変形ランレングス法の場合がランレングス法の場合より小さくなっており，データ圧縮効率の改善が得られることが確認される。なお，白ランの個数の差は，当然，漢字パターンの行パターン数と一致している。

また，符号化面積率は0.６２であった。

黒ランの個数が変形ランレングス法の白ランの個数より多いのは黒ランから始まる行パターンが存在するためであり，その個数は８００字について６２個であった。

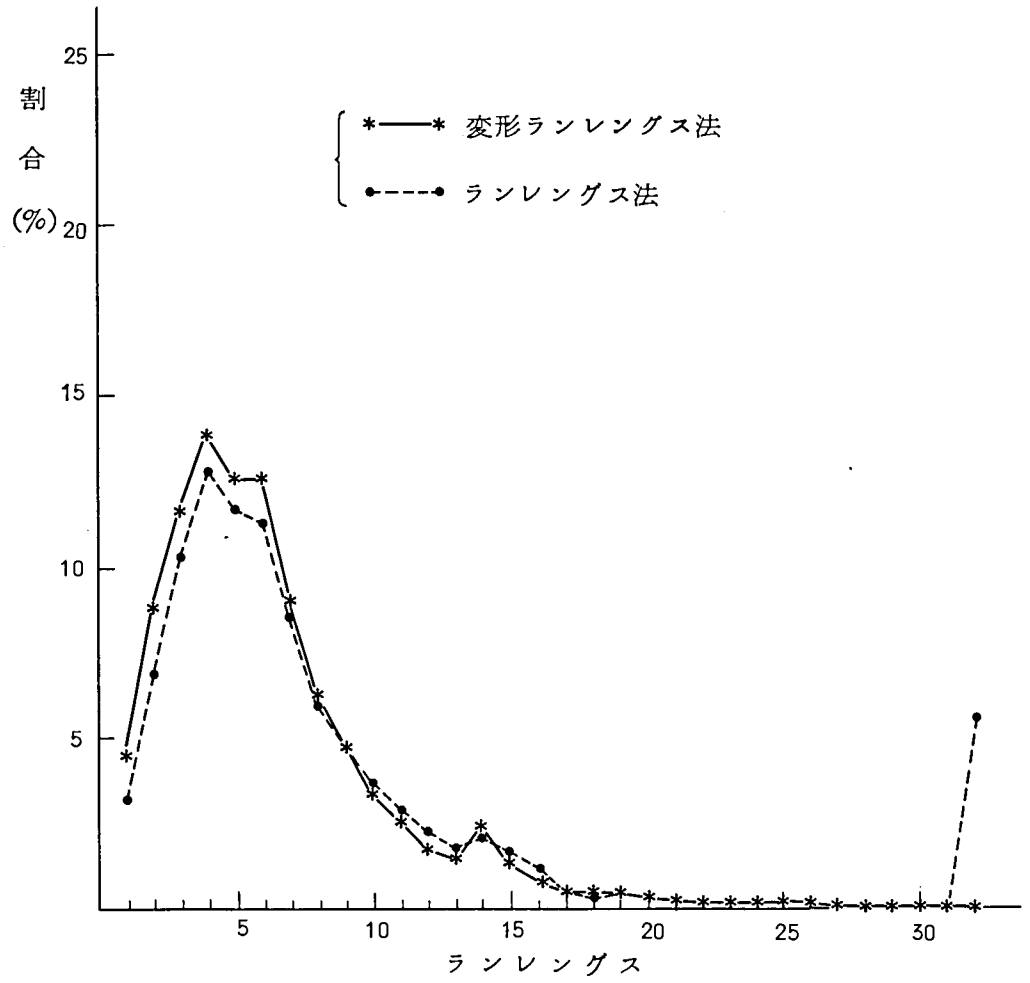


図 2.17　ランレングス法と変形ランレングス法の白ランレングス分布の比較

表 2.9　ランレングス分布データ
（試料：（32×32）漢字パターン800字）

（＊個数は1字当り）（エントロピーの単位：ビット）

項目 / 符号化法	白ラン				黒ラン				符号化面積率
	個数＊	平均長	分散	エントロピー	個数＊	平均長	分散	エントロピー	
ランレングス法	101.4	8.1	51.0	3.95	69.5	2.9	11.1	1.82	1.0
変形ランレングス法	69.4	6.2	15.7	3.75					0.62

表2.10は行パターン1個に含まれる黒ランの個数分布を示したものであり，最大9個の黒ランが含まれ，平均2.2個，エントロピーは2.51ビットであった。

表2.10　行当りの黒ランの個数分布
（（32×32）明朝体漢字パターン800字）

黒ランの個数	0	1	2	3	4	5	6	7	8	9	平均個数	エントロピー
個数分布	4,556	3,849	6,535	5,743	3,469	1,208	189	45	3	3	2.2	2.51bit
割合 (%)	17.8	15.0	25.5	22.4	13.6	4.7	0.7	0.2	0.0	0.0		

図2.18は変形ランレングス法での符号構成の概念図であり，黒ランの個数を示す個数データ部とランレングスデータ部から構成される。個数データ部には漢字パターンの上位の行パターンから順次，その中に含まれる黒ランの個数を示すデータ N_1，N_2 ……，N_m が格納されており，ランレングスデータ部には白ラン，黒ランの順で交互にランレングス WL_1，BL_1，……，WL_R，BL_R が格納されている。

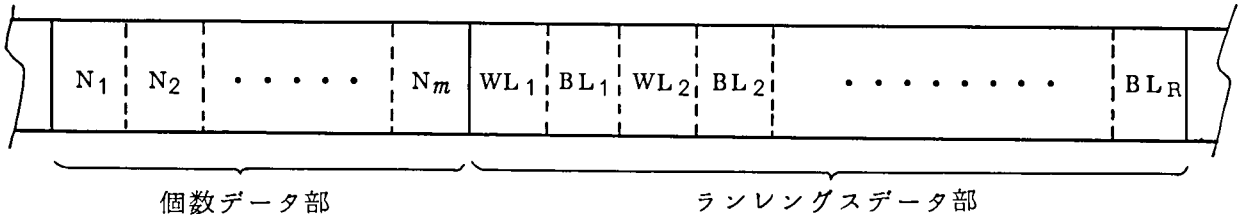


図2.18　変形ランレングス法の符号構成概念図

復号時には個数データ部内のデータを用いて，上位の行パターンから順次，その中に含まれる黒ランの個数 N_i（$i=1, 2, \cdots\cdots, m$）を知り，ランレングスデータ部内のランレングス符号を $2N_i$ 個ずつ区切ることで容易に復号することができる。

なお，黒ランから始まる行パターンに対しては，漢字パターン内に存在し得ない長さのダミーの白ランを含ませることで同一の復号処理が可能となる。

変形ランレングス法による，サイズ（$m \times m$）の漢字パターンの1個の符号量を $Q(m)$ とすると，黒ランの個数を示す個数指定符号量 $Q_1(m)$ とランレングス符号量 $Q_2(m)$ の和として

$$Q(m) = Q_1(m) + Q_2(m) \qquad (2.19)$$

で与えられる。

個数指定符号量$Q_1(m)$は，行パターン1個当りの平均符号量を$q_n(m)$とするとき

$$Q_1(m) = m \cdot q_n(m) \qquad (2.20)$$

となる。

また，ランレングス符号量$Q_2(m)$は白ラン，黒ランの平均符号量をそれぞれ，$q_R^W(m)$，$q_R^B(m)$，漢字パターン1個当りの平均の黒ランの個数を$N(m)$とすると，

$$Q_2(m) = N(m)(q_R^W(m) + q_R^B(m)) \qquad (2.21)$$

で与えられる。

表2.10と表2.9に示す実測値を式(2.20)と式(2.21)に代入して，データ圧縮率を求めると，

$m = 32$

$q_n(32) = 2.51$

$N(32) = 69.5$

$q_R^W(32) = 3.75$

$q_R^B(32) = 1.82$

であるから

$Q_1(32) \fallingdotseq 80.3$

$Q_2(32) \fallingdotseq 387.1$

$Q\ (32) \fallingdotseq 467.4$

となる。

データ圧縮比$C(m)$を

$$C(m) = \frac{m^2}{Q(m)} \qquad (2.22)$$

で定義するとき，$C(32) = 2.19$が得られる。

一方，従来の白黒分離ランレングス法では表2.8に示した値から，データ圧縮比$C(32) = 1.82$となる。

この結果，変形ランレングス法は，従来のランレングス法に比較して，データ圧縮比で約20%の改善が図られることになる。なお，黒ランから始まる行パターンに対して付加するダミーの白ランは800字(約55,500個の白ラン)中62個しか出現

しないため，通常の白ランとして上記のデータ圧縮比は算出した。

(2) 2次元符号化モデル（周辺符号化方式）

図形の2次元的な相関を利用する符号化方式として，ファクシミリ信号の帯域圧縮符号化方式である，変化点相対アドレス符号化方式（Relative Address Coding：RAC）[60]や，境界差分符号化方式（Edge Difference Coding：EDIC）[61]，また両者を組合せたREAD方式[62]など2次元逐次符号化方式がある。

これらの符号化方式は，いずれも，ファクシミリの主走査方向に見て，画素の状態が白から黒，あるいはその逆に変化する画素（変化画素とよぶ）間の距離を符号化するものであり，距離の計測の起点画素を，直前の走査線上の変化画素の位置を考慮して，符号化すべき距離が短かくなるように切り替えることにより，データ圧縮効果を高めた方式である。

したがって，これらの符号化方式では，変化画素間の距離の符号化に際しては，距離計測の起点画素を識別するための符号化モード情報を付加してやることが必要である。

上記の2次元逐次符号化方式の適用対象であるファクシミリ画像と本研究の対象である漢字パターンとを符号化の観点から比較すると以下の相違点があげられる。

① ファクシミリ画像の符号化では主走査および副走査方向共に1方向走査であり，符号化，復号化処理のための同一箇所の繰り返し走査は事実上不可能である。

一方，漢字パターンの符号化，復号化処理においては，メモリ上での処理であるため，自由な走査が可能である。

② 漢字パターンはファクシミリ画像に比べて，画像サイズが小さく，変化画素間の距離も短いため，符号化モード情報の負担能力が小さい。

そこで，繰り返し走査を行なうことにより符号化モード情報を無くした，次の周辺符号化方式を2次元符号化モデルとして設定する。

図2.19を用いて周辺符号化方式について述べる。

漢字パターンを構成する各ストロークを単位に符号化する方式である。すなわち，各ストロークの周辺に位置する2種類の周辺画素（ストロークの左，右の周辺上の周辺画素をそれぞれ左周辺画素，右周辺画素とよぶ）の位置を，そのストロークに属する直前の，かつ，同一種類の周辺画素の位置からのずれ，として一意的に定めることで，符号化モード情報を除去することとした。

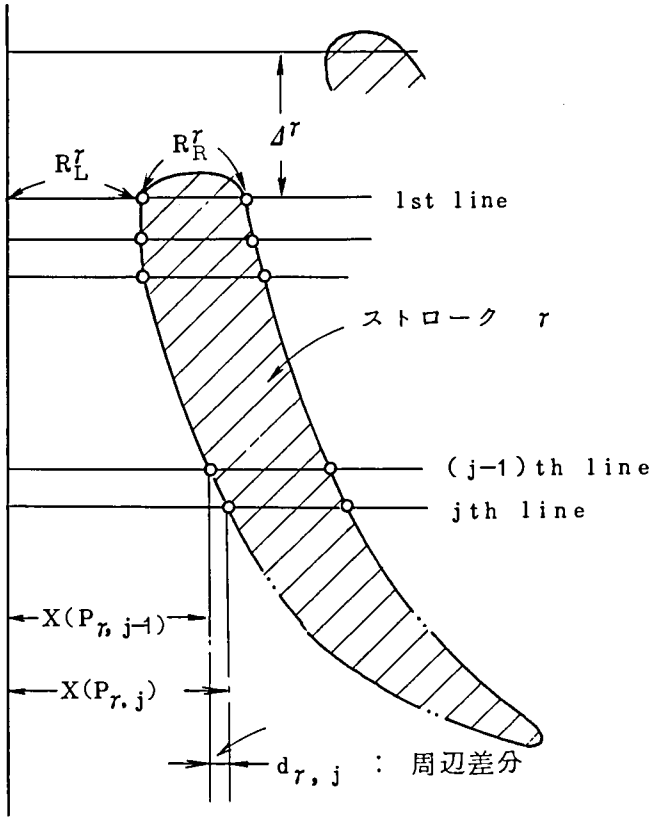


図 2.19　周辺符号化法の説明図

ストロークの漢字パターン内で占める位置は各ストローク上端の黒ランの位置をランレングス法で指定することによって行なう。

なお，図 2.20 (a) に示すように，1つのストローク内の隣接する行間に，複数個の同一種類の周辺画素（左周辺画素，右周辺画素）がある場合には，(b)に示すように複数個のストロークに分離し，それらの複数個の周辺画素の内，最も左側の周辺画素間のずれを符号化する。

今，ストローク r に属する行のうち，上から第 j 番目の行内の周辺画素の位置を $X(P_{r,j}^{*})$ とするとき，周辺差分 $d_{r,j}^{*}$ を

$$d_{r,j}^{*} = X(P_{r,j}^{*}) - X(P_{r,j-1}^{*}) \qquad (2.23)$$

で定義する。＊印は周辺画素，周辺差分の左，右の識別が必要なとき，それぞれ L，R とする。

以上述べた周辺符号化方式により，ストローク r 1個の符号量 Q_r は次式で与えら

れる。

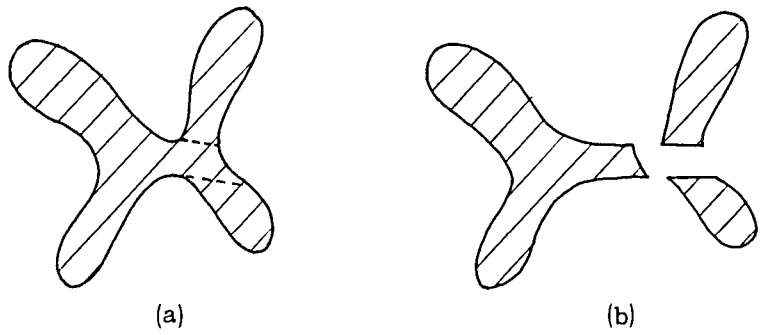


図 2.20　ストロークの分割

$$Q_\gamma = (q^H_{\gamma,R} + q^V_{\gamma,R}) + (N_\gamma - 1)(q^L_{\gamma,d} + q^R_{\gamma,d}) + q_{\gamma,S} \qquad (2.24)$$

ここで，$q^V_{\gamma,R}$，$q^H_{\gamma,R}$ はストローク γ の位置指定用の縦，横のランレングス Δ^γ と R^γ_L，R^γ_R の符号量，N_γ，$q_{\gamma,S}$ はストローク γ がまたがる行数とその記述符号量，$q^*_{\gamma,d}$ は周辺差分の平均符号量である。

したがってS個のストロークを含む漢字パターン1個の符号量Qは

$$Q = \sum_{\gamma=1}^{S} Q_\gamma \qquad (2.25)$$

となる。

今，サイズ（$m \times m$）の多数の漢字パターンの集合を考え，その平均としての漢字パターン1個の符号量をあらためて$Q(m)$と表わせば式（2.25）は次のようになる。

$$Q(m) = N(m)\{q^L_d(m) + q^R_d(m)\} + S(m)\Big[\{q^H_R(m) + q^V_R(m)\} - \{q^L_d(m) + q^R_d(m)\} + q_S(m)\Big],$$

$$q^H_R(m) = q^R_R(m) + q^L_R(m) \qquad (2.26)$$

ここで，$N(m)$，$S(m)$は漢字パターン1個当りの黒ランとストロークの平均個数である。また，式（2.24）で用いた各記号からサフィックスγを除いた記号は，それぞれ多数の漢字パターンに対して求めた各変数の平均値を意味するものとする。

表2.11に（32×32）の明朝体漢字パターン800個について求めた各変数の

実測値を示す。

表 2.1 1　周辺符号化用データ

項目	周辺差分				位置指定用ランレングス				列方向位置指定符号量q_R^V（ビット）	ストローク		
	左		右		左		右			個数 S	長さ	
	平均値	エントロピー q_d^L	平均値	エントロピー q_d^R	平均値	エントロピー q_R^L	平均値	エントロピー q_R^R			平均値	エントロピー q_S
実測値	0.57	1.74	1.39	2.54	6.2	3.89	1.7	1.52	2.58	11.3	5.1	3.4

周辺差分の平均値およびエントロピーが左，右で異なり，右周辺差分のそれらが大きな値となっているのは，前述したように，枝分れした形状のストロークに対して，左周辺差分が小さくなるようにストロークの分割を行なったためである。

図 2.2 1 に周辺符号化方式の符号構成の 1 例を示す。各漢字パターン毎に，その中に含まれるストローク数を指定するストローク数指定部とストローク記述部から成り，ストローク記述部内の各ストロークは位置指定部と周辺差分データ部で構成する。

ストロークの位置指定部にはストローク上端の黒ランの位置を記述するランレングスデータとストロークがまたがる行数を指示するストローク長データを格納しておく。

復号時には，ストロークの位置指定部内のランレングスデータを用いて，ストローク上端の黒ランを復元した後，ストローク長データを用いて所定の個数の周辺差分データを復号することによりストロークの周辺が復元される。最後に，左周辺画素と右周辺画素で挟まれた間の画素の状態を“黒”状態と書きなおすことにより漢字パターンが復号される。

表 2.1 1 に示した各変数の実測値を式（ 2.2 6 ）に代入して符号量 Q（ 3 2 ）とデータ圧縮比 C（ 3 2 ）を求めると次のようになる。

$$N(32) = 69.5 \qquad \text{（表 2.9 より）}$$

$$q_d^L(32) = 1.74$$

$$q_d^R(32) = 2.54$$

$$S(32) = 11.3$$

$$q_R^H(32) = q_R^R(32) + q_R^L(32)$$

$$= 5.41$$

$$q_R^V\ (32) = 2.58$$

$$q_S\ (32) = 3.4$$

であるから，式（2.26）から

$$Q\ (32) = 377.8$$

となり，データ圧縮比

$$C(32) = \frac{1,024}{Q(32)} \fallingdotseq 2.71$$

を得る。

前述のファクシミリ用の2次元逐次符号化方式の中で，符号化モード情報が最も少なくてすむＲＡＣ方式を取りあげ，周辺符号化方式との比較を行なう。

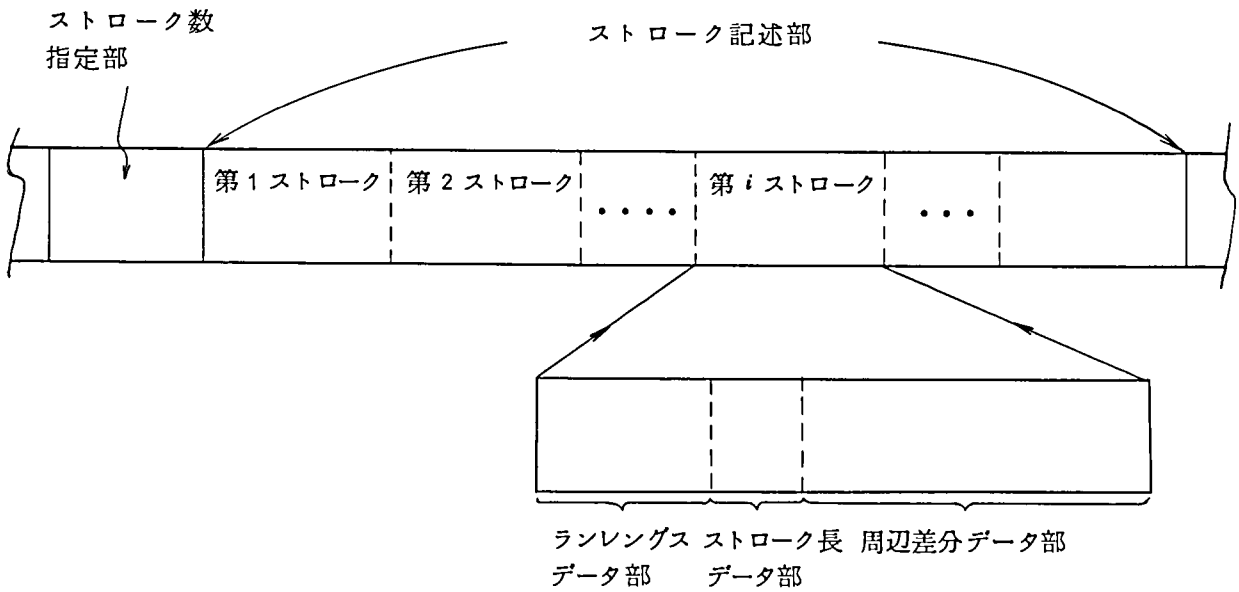


図2.21　周辺符号化方式の符号構成概念図

表2.12に表2.11と同じく，サイズ（32×32）の明朝体漢字パターン800字について求めたＲＡＣ方式用の実測データを示す。

ＲＡＣ方式では符号化対象である変化画素の位置を同一行内の直前の変化画素からの距離を用いて指定する水平モードと，1つ前の行内の変化画素からの距離を用いて指定する垂直モードの2つのモードがあるが，表2.12から分かるように，漢字パターンでは約71％の変化画素が垂直モードで符号化されている。

ＲＡＣ方式では，両符号化モードを識別するための符号化モード情報を各変化画素

毎の符号の中に含ませ，変化画素間の距離はモディファイド・ハフマン符号化法等を用いて符号化が行なわれるが，ここでは，周辺符号化方式と情報量の理論値での比較を行なうために，符号化モードおよび正，負の符号が異なる符号化距離は全て異なる事象と見なして，１個の変化画素の符号化に要する情報量を求めた。その結果を表2.１２では一括モードとして示している。

表2.１２　ＲＡＣ方式用データ

項目＼符号化モード	垂直モード	水平モード	一括モード
符号化画素数	82,199	33,485	115,684
平均符号化距離（絶対値）	0.93	6.51	2.55
エントロピー*	1.86	2.90	3.03

＊　正，負の異なる符号化距離は別事象として算出

この結果，ＲＡＣ方式による漢字パターン１個当りの符号量は，

$$Q(32) = \frac{3.03 \times 115,684}{800}$$

$$\fallingdotseq 438.2$$

となり，データ圧縮比

$$C(32) = \frac{1,024}{Q(32)}$$

$$\fallingdotseq 2.33$$

となる。

したがって，周辺符号化方式はＲＡＣ方式に比較して，データ圧縮比で約１６％高い圧縮効率が得られることが分かった。

2.3.2　漢字パターンのサイズとデータ圧縮比の関係に対する考察

本項ではサイズの異なる各種の漢字パターンに対する圧縮比を予測することを目的に，前項で述べた２つの符号化モデルを対象に漢字パターンのサイズとデータ圧縮比の関係について考察する。

(1) 理論的考察

今，サイズ（$m_0 \times m_0$）の漢字パターンを基準サイズの漢字パターンとして，サイズがk倍の（$m \times m$）の漢字パターンのデータ圧縮比について考察する。

サイズの変化に対して，ほぼ相似的に漢字パターンが変化することを前提とすれば，近似的に次式が成立する。

$$q_n(n) = q_n(m_0) \qquad (2.27)$$

$$S(m) = S(m_0) \qquad (2.28)$$

$$N(m) = k \cdot N(m_0) \qquad (2.29)$$

但し

$$m = k \cdot m_0 \qquad (2.30)$$

であり，$q_n(m)$はサイズ数（$m \times m$）の漢字パターンに対する，行パターン当りの黒ランの個数指定のための符号量，$S(m)$，$N(m)$はそれぞれ漢字パターン1個当りのストローク数と黒ランの個数である。

すなわち，パターンの相似的な変換に対して，行パターンに含まれる黒ランの個数分布と漢字パターン内のストローク数は不変であり，パターン内の黒ランの総個数はサイズmに比例して増加することになる。

そこで以下，変形ランレングス法と周辺符号化方式で，主に，符号量を支配するランレングスの平均符号量と周辺差分の平均符号量のサイズに対する変化について検討する。

(i) ランレングス符号量の変化

漢字パターンは縦直線を多数含んでおり一般の図形と比べ，黒ランレングス分布が縦直線幅に対応する部分に高いピークを持つ特殊な図形であると考えられる。

そこで，図2.22に示すように黒ランレングス分布$f(L_i)$を縦直線に含まれる黒ランのランレングス分布$f_V(L_i)$と，それ以外の横直線，斜線，曲線に含まれる黒ランのランレングス分布$f_H(L_i)$とに分けて考察する。

すなわち，

$$f(L_i) = f_H(L_i) + f_V(L_i) \qquad (2.31)$$

$$f_V(Li) = \begin{cases} 0 & : \text{for} \quad L_i \neq L_W \\ N_V & : \text{for} \quad L_i = L_W \end{cases}$$

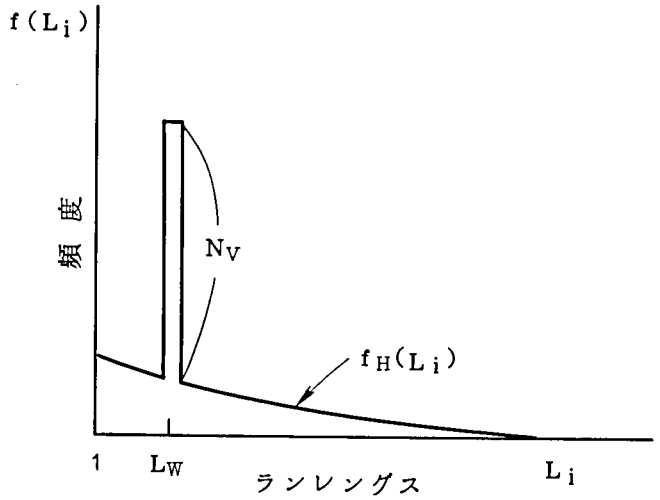


図 2.2.2　漢字パターンの黒ランレングスモデル

とする。ここで L_W は縦直線の線幅である。

この時，黒ラン 1 個のエントロピーは

$$E = -\sum_i p(L_i) \cdot \log_2 p(L_i) \qquad (2.32)$$

$$p(L_i) = \frac{f(L_i)}{T}, \quad T = \sum_i f(L_i) \qquad (2.33)$$

で与えられる。

漢字パターンに対して $f_H(L_W) \ll f_V(L_W)$ が成立するため，式 (2.32) は次のように変形される。

$$E \simeq -p'(L_W) \cdot \log_2 p'(L_W) - \sum_{L_i \neq L_W} p(L_i) \log_2 p(L_i) \qquad (2.34)$$

但し

$$p'(L_W) = \frac{N_V}{T} \qquad (2.35)$$

である。

今，漢字パターンのサイズ m が基準サイズ m_0 を k 倍して得られるものとすれば，

$$T = k \cdot T_0$$

$$L_W = k \cdot L_{W_0}$$

$$N_V = k \cdot N_{V_0}$$

が成立する。但し，T_0 ，L_{W_0}，N_{V_0} はそれぞれ基準サイズ（$m_0 \times m_0$）の漢字パターンに対して得られる黒ランの総数，縦直線の線幅，縦直線に含まれる黒ランの個数である。

したがって式（2.35）の p'（L_W）はサイズの変化に対して不変量となり，その結果，式（2.34）の右辺第1項もまた不変量となる。

また，一方，相似な漢字パターンの変化に対する分布 f_H の変化を次式で近似する。

$$f_H(L_j) = f_{H_0}(L_i) \qquad (2.36)$$

$$i = [\,j/k\,]$$

$$\begin{cases} i = 1,\ 2,\ \cdots\cdots,\ m_0 \\ j = 1,\ 2,\ \cdots\cdots,\ km_0 \end{cases}$$

但し〔 〕はガウス記号であり，$f_{H_0}(L_i)$ はサイズ（$m_0 \times m_0$）の基準漢字パターンに対する分布である。

この時，式（2.34）は，付録2.2に示すように，

$$E = E_0 + \left(1 - \frac{N_{V_0}}{T_0}\right) \cdot \log_2 k \qquad (2.37)$$

と変形される。

一方，白ランについては，黒ランの場合のようにランレングス分布に鋭いピークは現われないため，黒ランに対して求めた式（2.37）で $N_{V_0} = 0$ とおくことにより，

$$E = E_0 + \log_2 k \qquad (2.38)$$

が得られる。

黒ランは，ランレングス分布自体が，ランの発生が単純マルコフ過程に従う場合にとる幾何分布と著しく異なっており，ラン当りの情報量の増加は一般図形に対するものと異なる。

一方，ランの発生が単純マルコフ過程に従うものとすれば，サイズ m の変化に対するラン当りのエントロピー $E(m)$ の変化は，付録2.3に示すように，

$$E = E_0 + \log_2 k + \Delta\left(k,\ \widehat{L}(m_0)\right)$$

$$\Delta(k, y) = \log_2 \frac{(1-\frac{1}{y})^{y-1}}{(1-\frac{1}{k \cdot y})^{k y-1}} \qquad (2.39)$$

で与えられる。

但し，$\hat{L}$は漢字パターン内でのランレングスの平均値である。

図2.23にyをパラメータとして$\Delta(k, y)$のグラフを示す。

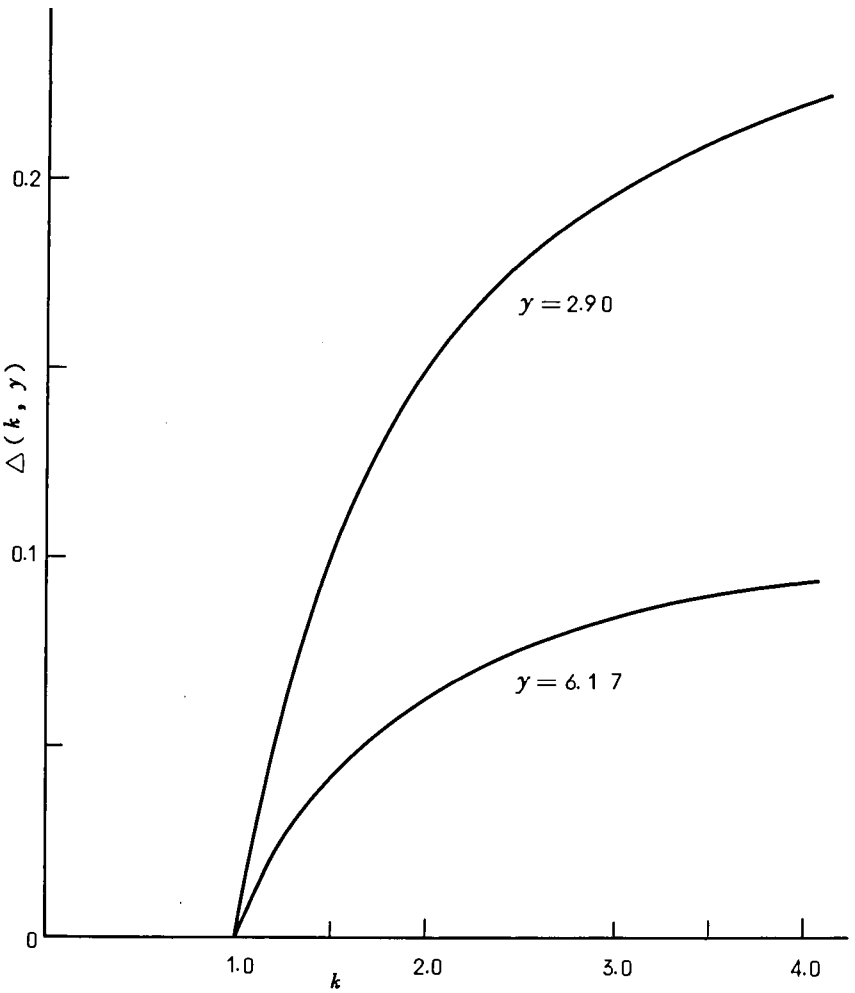


図 2.23　$\Delta(k, y)$ の変化

式（2.37），（2.38）それに式（2.39）を用いて，本研究で設定した漢字パターンに対するランレングス分布モデルから求めたランのエントロピーと単純マルコフ過程を基にして求めたランのエントロピーを比較すると，

白ランに対しては，

$$\delta W(k\cdot m_0) = -\Delta(k,\ \hat{L}_W(m_0)) \qquad (2.40)$$

黒ランに対しては，

$$\delta_B(k\cdot m_0) = -\frac{N_{V_0}}{T_0}\log_2 k - \Delta(k,\ \hat{L}_B(m_0)) \qquad (2.41)$$

となる。$\hat{L}_W$，$\hat{L}_B$ はそれぞれ白ラン，黒ランの平均ランレングスである。

図2.23に示した $\Delta(k, y)$ のグラフから分かるように，平均ランレングスが長い白ランに対しては両者の差 δ_W は小さいが，黒ランに対しては両者に大きな差が生じることになる。

(ii) 周辺差分符号量の変化

構成線分の方向が均等である一般的な図形の相似変換に対して，周辺差分の符号量は不変量と考えられる。

そこで，前節で述べた周辺符号化方式で符号化の単位としたストロークのモデルを設定して，漢字パターンのサイズと周辺差分符号量の関係を考察する。

今，単純なストロークのモデルとして図2.24に示すような，長さがそれぞれ L_{V_0} と L_{S_0} の縦直線と傾きが45度の斜線から成る線分に異なる長さの N 本の横直線が交叉しているストロークを考える。

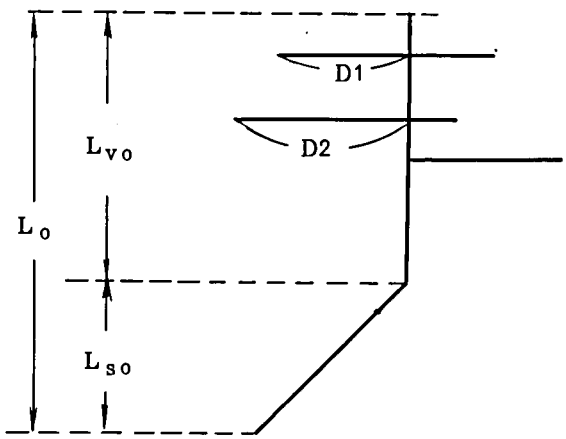


図2.24　ストロークモデル

このストロークに対する左周辺差分の分布 $f(d)$ は

$$f(d) = \begin{cases} L_{V_0} - 2N & : \text{for } d = 0 \\ L_{S_0} & : \text{for } d = 1 \\ 1 & : \text{for } d = \pm D_1 \\ \vdots & \\ 1 & : \text{for } d = \pm D_N \end{cases} \qquad (2.42)$$

となる。但し，$D_1 \sim D_N$ は N 本の横直線によって生じる周辺差分である。

したがって周辺差分のエントロピーは

$$E_0 = -\frac{L_{V_0}-2N}{L_0}\cdot\log_2\frac{L_{V_0}-2N}{L_0}-\frac{L_{S_0}}{L_0}\log_2\frac{L_{S_0}}{L_0}-\frac{2N}{L_0}\cdot\log_2\frac{1}{L_0}$$

$$= -(C_V-C_N)\log_2(C_V-C_N)-C_S\cdot\log_2 C_S+C_N\log_2 L_\phi \qquad (2.43)$$

但し，

$$L_0 = L_{V_0}+L_{S_0}$$

$$C_V = \frac{L_{V_0}}{L_0}$$

$$C_S = \frac{L_{S_0}}{L_0}$$

$$C_N = \frac{2N}{L_0}$$

となる。

今，漢字パターンのサイズがk倍になり，パターンが相似的に変化したとすると，C_V，C_Sは不変量であり，C_Nは$1/k$となる。したがって，エントロピーは次のようになる。

$$E(k) = -(C_V-\frac{C_N}{k})\cdot\log_2(C_V-\frac{C_N}{k})-C_S\cdot\log_2 C_S$$

$$+\frac{C_N}{k}\log_2(k\cdot L_0) \qquad (2.44)$$

また，基準サイズm_0の場合に対するエントロピーの相対的変化$\Gamma_E(k)$は次式で与えられる。

$$\Gamma_E(k) = \frac{E(k)}{E(1)} \qquad (2.45)$$

水平直線が無い場合には$C_N = 0$となり，$E(k)$は定数となり，サイズの変化に対して周辺差分のエントロピーは不変量となる。

図2.25に$L_0 = 6$，$C_V = 0.6$，$C_S = 0.4$の場合について，横線の含有率とも云うべきC_Nをパラメータとして$\Gamma_E(k)$の変化の様子を示す。周辺差分のエントロピーはサイズの変化倍率kの増加に対して単調に減少し，その割合は横直線が多数含まれる程大きいことが分かる。

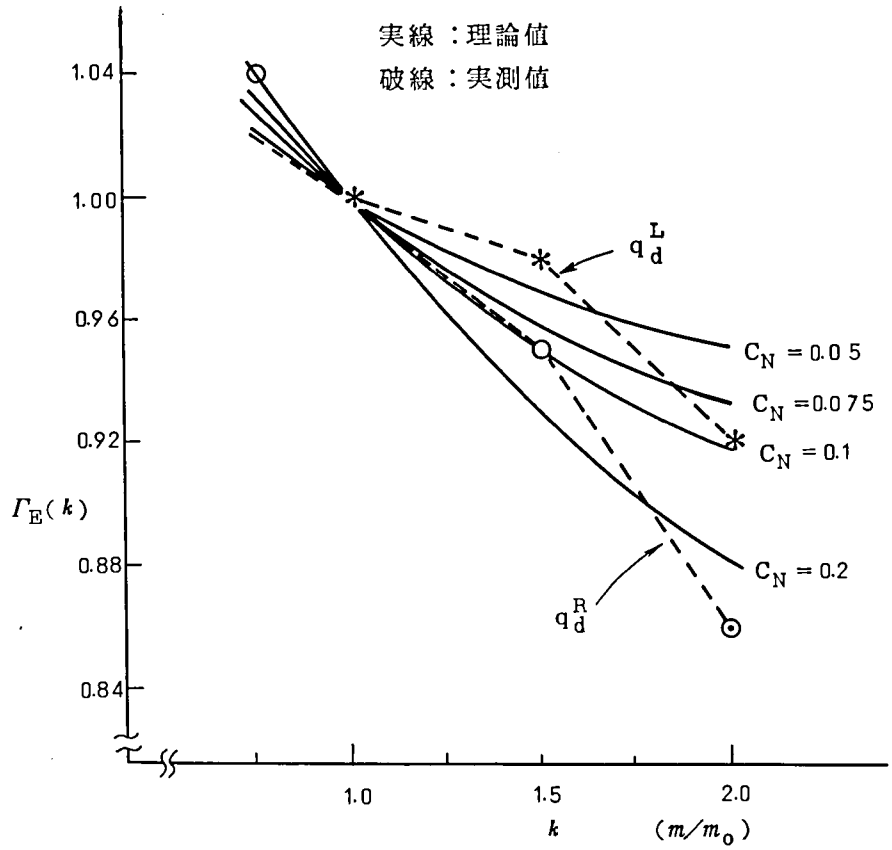


図 2.25　周辺差分エントロピーの相対変化 $\Gamma_E(k)$

(2)　実験的検証

付録 2.4 に示した 40 種の漢字についてサイズが（24×24），（32×32），（48×48）および（64×64）の 4 種類の漢字パターンを作成し，それらの漢字パターンを対象にサイズの変化による各変数の変化を調べた。その結果を表 2.13 に示す。同表で各変数の値はサイズ（32×32）の漢字パターンの各変数値に対する相対値である。

サイズ（32×32）は明朝体を含む漢字パターンの自然な表現に最低必要となるサイズと考え，以下，これを基準サイズとする。

表 2.13 から，サイズの変化に対する各変数の変化のタイプは，大きく次の 3 つに分かれることがわかる。

①　サイズの変化に対して不変な変数

- 漢字パターン 1 個当りのストローク数 S
- 行パターン内の黒ランの個数指定符号量 q_n

表 2.13　サイズと各変数の関係（試料数 40 字）

サイズ変化率 k	1字当りの黒ランの個数 N	ストローク数 S	変形ランレングス法					周辺符号化法							
			黒ラン数指定符号量 q_n	白ラン		黒ラン		左差分		右差分		白ラン		黒ラン	
				平均長 L_R^W	エントロピー q_R^W	平均長 L_R^B	エントロピー q_R^B	平均値	エントロピー q_d^L	平均値	エントロピー q_d^R	平均値 L_R^L	エントロピー q_R^L	平均値 L_R^R	エントロピー q_R^R
0.75	0.74	0.98	0.97	0.68	0.85	0.99	0.97	0.97	1.02	0.98	1.04	0.74	0.86	1.18	1.21
1.0	1.0														
1.5	1.50	1.00	1.01	1.44	1.14	1.66	1.31	1.03	0.98	1.01	0.95	1.57	1.16	1.47	1.43
2.0	2.00	1.03	0.99	1.98	1.23	2.04	1.37	1.00	0.92	0.98	0.86	2.06	1.22	2.06	1.50

・左，右の周辺差分の平均値

② サイズに比例して変化する変数

・漢字パターン1個当りの黒ランの個数 N

・白ラン，黒ランのランレングスの変数値。（$k=0.75$，すなわち，サイズ（24×24）の黒ランについては，明朝体を表現するため，縦線幅を，（32×32）と同じく，2ドットとしたため，比例関係から外れている。）

③ サイズの変化に対して複雑に変化する変数

・白ラン，黒ランのエントロピー q_R^W，q_R^B

・左，右の周辺差分のエントロピー q_d^L，q_d^R

この結果，式（2.27）～式（2.29）の各式が近似的に成立することが実験的に確認された。

また，図2.25に実測した周辺差分のエントロピーの基準サイズに対する相対変化 $\Gamma_E(k)$ を破線で示している。左周辺差分，右周辺差分はそれぞれ，式（2.44）で $C_N=0.075$ と $C_N=0.2$ 程度の値を設定した場合で，ほぼ近似できることがわかる。試みにこれらの値を用いてストロークモデルの形状を考えれば，サイズ（32×32）の画素マトリクス上で，長さ6画素の縦直線と斜線に左側に0.2本，右側に0.6本の横直線が出たストロークとなる。

ストローク形状についての定量的な検証は行なっていないが，右側の横直線の本数が左側のそれに比較して多くなる点では，周辺符号化方式の符号化アルゴリズムと定性的に一致することがわかる。

図2.26にランレングスのエントロピーの変化について式（2.37），（2.38）の理論値を実線で，実測値を破線で示している。但し，両者とも基準サイズとした（32×32）の漢字パターンの値で正規化して示している。

同図から白ランに対しては式（2.38）の理論値と実測値が良く一致することがわかる。

また，黒ランについては式（2.37）で $N_{V_0}/T_0=0.3$ 程度でほぼ一致することが分かる。この結果，従来，ファクシミリ信号の帯域圧縮の議論で一般図形を対象に仮定される単純マルコフモデルは漢字パターンに対しては必ずしも精度良く成立しないことになり，漢字を含む日本語文書を対象とした帯域圧縮の検討では考慮する必要がある。

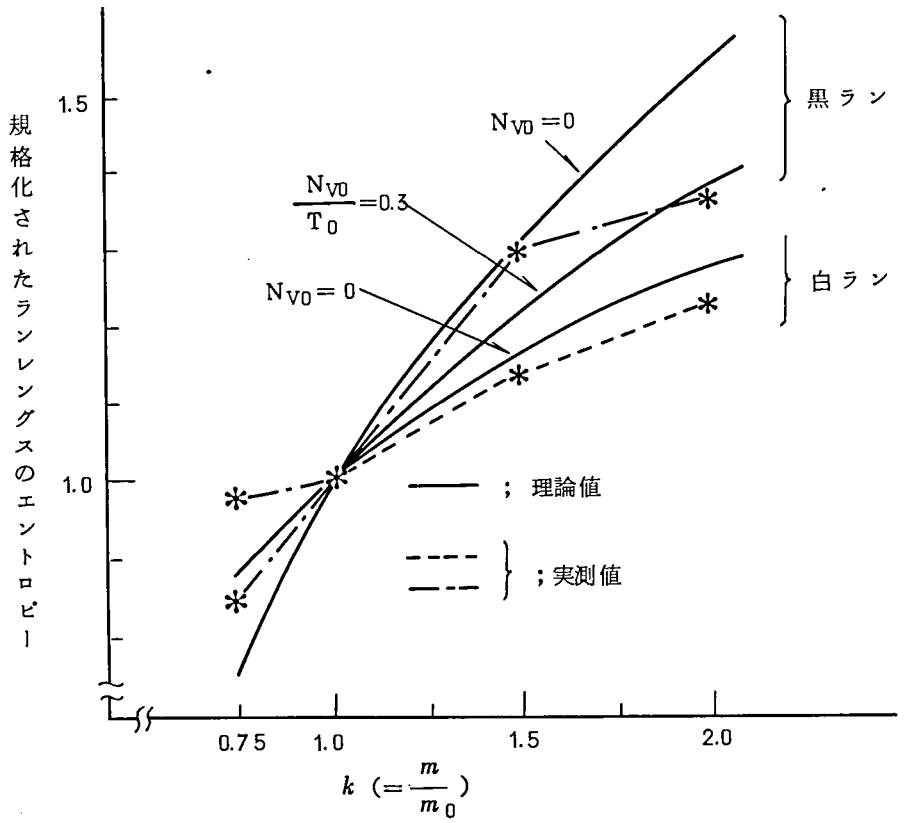


図 2.2 6　ランレングス符号量の相対変化

(3)　サイズとデータ圧縮比

(i)　変形ランレングス法

変数 $q_n(m)$，$N(m)$，$q_R^B(m)$，$q_R^W(m)$ がサイズ m の変化に対して，それぞれ式（2.2 7），（2.2 9），（2.3 7）および（2.3 8）に従って変化することから，これらを式（2.1 9）～（2.2 1）に代入して，漢字パターン1個当りの符号量は，サイズの変化倍率 k の関数として

$$Q(k\cdot m_0)=Q(m_0)\cdot k+N(m_0)\left(2-\frac{N_{V_0}}{T_0}\right)\cdot k\cdot\log_2 k \qquad (2.46)$$

で与えられる。

この結果，基準サイズ漢字パターンに対する相対的データ圧縮比 $\Gamma_C^R(k)$ は

$$\Gamma_C^R(k)=\frac{C(k.m_0)}{C(m_0)}=\frac{k}{1+N(m_0)\left(2-\frac{N_{V_0}}{T_0}\right)\cdot\log_2 k\,/\,Q(m_0)} \qquad (2.47)$$

となる。

式（2.47）に変数の実測値を代入すると漢字パターンに対するデータ圧縮比の予測式として次式を得る。

$$\Gamma_C^R(k)=\frac{k}{1+0.253\cdot\log_2} \qquad (2.48)$$

また，ランの発生が単純マルコフ過程に従うとした場合のデータ圧縮比を求めると次のようになる。

先ず，漢字パターン1個当りの符号量$Q(m)$は

$$Q(m)=m\cdot q_n(m)+N(m)\{\hat{L}_W(m)\cdot h(\hat{L}_W(m)) +\hat{L}_B(m)\cdot h(\hat{L}_B(m))\} \qquad (2.49)$$

となる。

$q_n(m)$，$N(m)$は，それぞれサイズmの漢字パターンに対する，行パターン内の黒ランの個数指定符号量と黒ランの総数であり，$h(\hat{L}_W(m))$，$h(\hat{L}_B(m))$はそれぞれ，平均ランレングスが$\hat{L}_W(m)$，$\hat{L}_B(m)$の白，黒ランの構成画素当りのエントロピーであり，

$$h(y)=-\frac{1}{y}\cdot\log_2\left(\frac{1}{y}\right)-\left(1-\frac{1}{y}\right)\log_2\left(1-\frac{1}{y}\right) \qquad (2.50)$$

である。（付録2.3）

式（2.27），（2.29）を式（2.49）に代入して

$$\hat{L}(k\cdot m_0)=k\cdot\hat{L}(m_0)$$

とすると，

$$Q(m)=k\cdot Q(m_0)+k\cdot N(m_0)\{2\cdot\log_2 k+\Delta(k,\hat{L}_W(m_0))+\Delta(k,\hat{L}_B(m_0))\}$$

となる。但し，$\Delta(k,y)$は式（2.39）に示すとおりである。

実測値を用いて，（32×32）の漢字パターンでの圧縮比で正規化した相対的データ圧縮比$\Gamma_C^{R'}(k)$は

$$\Gamma_C^{R'}(k)=\frac{k}{1+0.149\{2\cdot\log_2 k+\Delta(k,6.2)+\Delta(k,2.9)\}} \qquad (2.51)$$

となる。

図2.27に$\Gamma_C^R(k)$の変化を実線で示す。また，同図に，白ラン，黒ランのランレングス分布が単純マルコフ過程に従うとした場合の圧縮比の変化$\Gamma_C^{R'}(k)$を1点鎖線で示す。また，実際の漢字パターンに対して求めた圧縮比を同図に併せて示す。式（2.48）の理論式と良く一致することが分かる。

図2.27から圧縮比の変化を1次式に従うと見なして，その勾配の概略値を求めると約0.6となる。

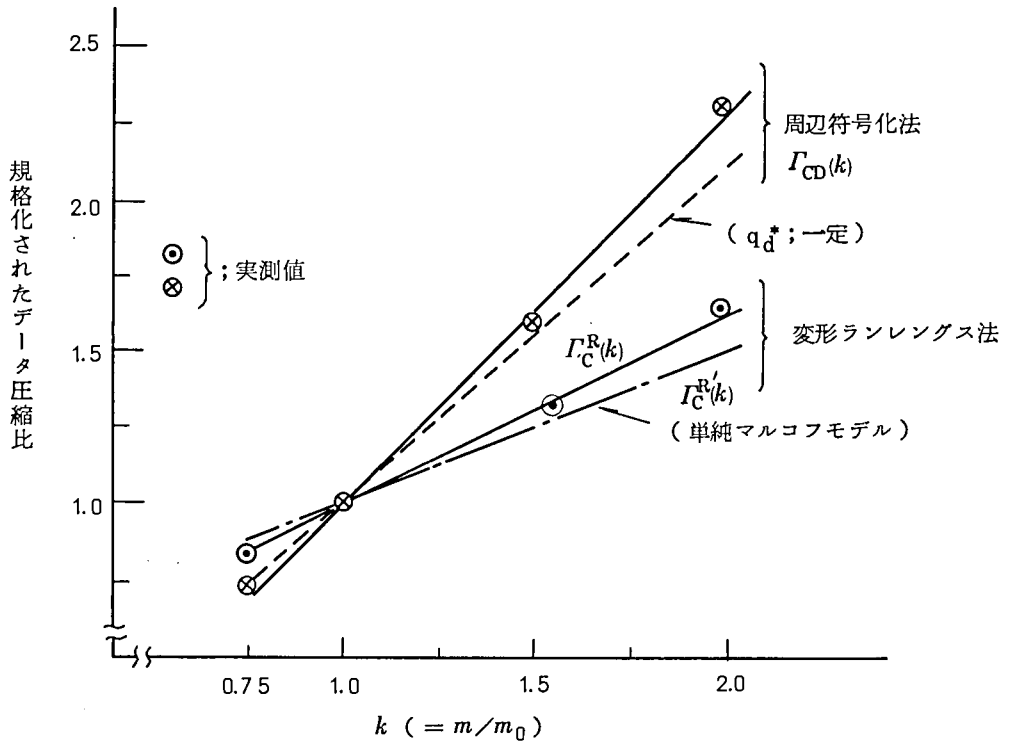


図2.27　データ圧縮率の相対変化

(ii)　周辺符号化方式

符号量を与える式（2.226）で，q_R^L，q_R^V，q_Sは式（2.38）に，q_R^Rは式（2.37）（$N_{V_0}/T_0=0.3$の場合）に従うものとし，また，周辺差分符号量は次式で変化するものとする。

$$q_d^*(m)=q_d^*(m_0)\Gamma_E^*(m/m_0) \qquad (2.52)$$

（＊はL又はR）

ここで，$\Gamma_E^*(k)$は式（2.45）で定義される。

この時，符号量は k の関数として次のようになる。

$$Q(k\cdot m_0)=k\cdot N(m_0)\{q_d^L(m_0)\cdot\Gamma_E^L(k)+q_d^R(m_0)\cdot\Gamma_E^R(k)\}$$
$$+S(m_0)[\{q_R^L(m_0)+q_R^R(m_0)+q_R^V(m_0)+q_S(m_0)$$
$$+(4-\frac{N_{V_0}}{T_0})\cdot\log_2 k-\{q_d^L(m_0)\cdot\Gamma_E^L(k)+q_d^R(m_0)\Gamma_E^R(k)\}] \quad (2.53)$$

表2.11に示すサイズ（32×32）の漢字パターンの実測値を上式に代入することにより，符号量は次式で与えられる。

$$Q(k\cdot m_0)=69.5k\{1.74\,\Gamma_E^L(k)+2.54\,\Gamma_E^R(k)\}$$
$$+11.3[(11.4+3.7\log_2 k)-\{1.74\,\Gamma_E^L(k)+2.54\,\Gamma_E^R(k)\}] \quad (2.54)$$

基準サイズ（32×32）の漢字パターンに対して得られるデータ圧縮比を基準にした相対的な圧縮比の変化 $\Gamma_{CD}(k)=C(m)/C(m_0)$ の様子を図2.27に示す。

なお，同図に周辺差分の符号量 q_d^* を一定とした場合の変化を破線で示している。周辺差分符号量 q_d^* を一定とした場合にはサイズが大きい漢字パターンに対して実測値からのずれが大きくなり，不適切であることが確認される。

図2.27から，$0.75\leqq k\leqq 2$ の範囲で，圧縮比の変化が1次式に従うものと見なしてその勾配の概略値を求めると約1.3となる。

2.4 まとめ

2章では漢字パターン発生器の経済化の手段の1つとして漢字パターンのデータ圧縮法について考察した。

漢字パターンのデータ圧縮法は圧縮効率のみならず，復号処理に伴なう漢字パターンの発生速度の低下と復号装置の規模も考慮して，漢字パターン発生器の適用システムに応じて評価しなければならない。

本研究では，まず高価な漢字パターン発生器を多数のラスタ走査形漢字出力装置で共用するシステムへの適用を前提として，データ圧縮法を考察した。

その結果，

(1) 漢字パターンを，出力装置の走査方向と方向が一致する１次元部分パターン（行パターン）へと分割し，出現頻度に応じて行パターンへの符号割当てを行ないデータ圧縮効果を得る行パターン合成符号化法を提案した。

行パターン合成符号化法のデータ圧縮効率の上限として，（１８×１６），（２４×２４），（３２×３２）の漢字パターンに対して，それぞれ７７％，６６％，４９％を求めた。

また，上記の前提条件を満たす代表的な符号化法であるランレングス符号化法と比較して，データ圧縮率で５～１１％高い圧縮効果をもち，漢字パターン発生速度も数倍速いことを示した。

(2) 従来の部分パターンへの分割符号化法の検討では部分パターンの大きさの最適化に対する考察はなされていなかったが，部分パターン自体のデータ量と部分パターンの符号化データ量の両者を考慮することによって，データ圧縮効果を最大とする部分パターンの大きさの最適値が存在することを示した。

次に，漢字パターンのサイズとデータ圧縮比の関係について定量的に考察した。

その結果，

(3) １次元符号化モデルとして変形ランレングス法を，また，２次元符号化モデルとして周辺符号化法を提案して，（３２×３２）の明朝体漢字パターンに対して，データ圧縮比がそれぞれ，2.19，2.71となることを示した。

変形ランレングス法はランレングス法に比べて２０％，周辺符号化法はＲＡＣ符号化法に比べて１６％高い圧縮比を与えることになる。

(4) 漢字パターンのサイズの変化に対する，ラン当りのエントロピーの変化について考察し，従来ファクシミリ信号の帯域圧縮処理の研究で用いられてきた単純マルコフ過程に従うランの発生モデルは漢字パターンに対しては精度良く成立しないことを示した。

また，漢字パターン用のランレングス分布モデルを作成し，パターンサイズの変化に対するラン当りのエントロピーの変化式を求めた。

(5) 縦横直線を多数含む漢字パターンに対して，ストロークモデルを作成し，漢字パターンのサイズと周辺差分のエントロピーの関係式を求めた。

(6) 前記２つの符号化モデルについて，漢字パターンのサイズとデータ圧縮比の定量的関係式を求めた。

すなわち，（２４×２４）～（６４×６４）程度の範囲でサイズとデータ圧縮比は１次式で近似でき，その勾配は（３２×３２）の漢字パターンの圧縮比で正規化したとき，変形ランレングス法と周辺符号化法で，それぞれ，0.6と1.3となることを示した。

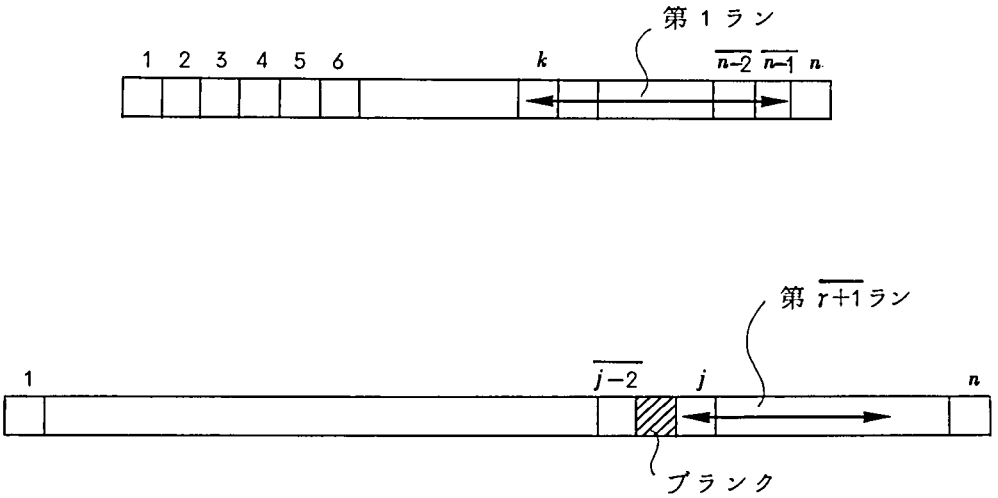
付録2.1　式（2.12）の証明

数学的帰納法を用いて証明される。

$$N(n, r)=\frac{1}{(2r)!}(n-2\cdot\overline{r-1})\cdots\cdot n\cdot(n+1) \qquad (付2.1.1)$$

(i)　$r=1$ の場合

（1行×n列）の行パターン内に1個の黒ランを任意の位置に任意の長さで配置する方法の数は付図2.1を参照して次のようになる。



付図2.1　黒ランの個数分布算出の参照図

始点がk（$1\leq k\leq n$）の黒ランの配置の方法$N(n, 1; k)$は

$$N(n, 1; k)=n-(k-1) \qquad (付2.1.2)$$

である。

始点kを1からnまで変えることによって1個の黒ランを配置する方法の数$N(n, 1)$は，

$$N(n, 1)=\sum_{k=1}^{n} N(n, 1; k)$$

$$=\sum_{k=1}^{n}(n+1-k)$$

$$=\frac{1}{2}n\cdot(n+1) \qquad (付2.1.3)$$

となり，$r=1$ で式（付 2.1.1）は成立する。

(ii) $r=r$ で式（付．2.1.1）が成立すると仮定する。

付図 2.1 を参照して，第（$r+1$）番目の黒ランを始点 j として，j～n の配置をする場合を考える。

今，第（$r+1$）番目の黒ランを最も右端に追加配置する場合を考えるが，そうでない場合には，一旦，（$r+1$）個の黒ランを配置した後，最も右端の黒ランを第（$r+1$）番目の黒ランとして配置したものと考えることにより，以下の議論は一般性を失わない。

先ず，第 r 番目までの黒ランを 1～$j-2$ までの（$j-2$）個のます目に配置する方法の数は式（付 2.1.1）より

$$\frac{1}{(2r)!}\{(j-2)-2(r-1)\}\cdot\{(j-2)-2(r-1)+1\}\cdots\cdots(j-2)\cdot(j-1)$$

である。

第（$r+1$）番目の黒ランの配置方法の数は式（付 2.1.2）より（$n-j+1$）である。第（$r+1$）番目の黒ランの始点 j がとり得る範囲は $\overline{2\cdot r+1}$ から n までである。

したがって，長さ n の行パターン内に（$r+1$）個の黒ランを配置する方法の数 $N(n, r+1)$ は

$$N(n, r+1)=\sum_{j=2r+1}^{n}\left[\frac{1}{(2r)!}\{(j-2)-2(r-1)\}\cdot\{(j-2)-2(r+1)+1\}\cdot\right.$$
$$\left.\cdots\cdots(j-2)\cdot(j-1)\cdot(n-j+1)\right]$$

となる。上式は更に次のように変形される。

$$N(n, r+1)=\sum_{j=2r+1}^{n}\left[\frac{1}{(2r)!}(j-2r)(j-2r+1)\cdots\cdots(j-2)\cdot(j-1)\cdot\right.$$
$$\left.(n-j+1)\right]$$

$J=j-2r$ とおいて

$$N(n, r+1)=\sum_{J=1}^{n-2r}\left[\frac{1}{(2r)!}\cdot J\cdot(J+1)(J+2)\cdots\right.$$
$$\left.\cdots(J+2r-2)(J+2r-1)\cdot(J+2r-n-1)\right]$$

$$=-\frac{1}{(2r)!}\sum_{J=1}^{n-2r}\cdot J\cdot(J+1)\cdot(J+2)\cdots(J+2r)$$

$$+\frac{(n+1)}{(2r)!}\sum_{J=1}^{n-2r} J\cdot(J+1)\cdot(J+2)\cdots(J+2r-1)$$

こゝで積和の公式[63]

$$\sum_{r=1}^{n} r\cdot(r+1)\cdots(r+k)=\frac{1}{k+2}\cdot\frac{(n+k+1)!}{(n-1)!} \qquad (付\ 2.1.4)$$

を用いると，$N(n, r+1)$は更に次のように変形される。

$$N(n, r+1)=\frac{(n+1)}{(2r)!}\cdot\frac{1}{2r+1}\cdot\frac{(n-2r+2r-1+1)!}{(n-2r-1)!}$$

$$-\frac{1}{(2r)!}\cdot\frac{1}{2r+2}\cdot\frac{(n-2r+2r+1)!}{(n-2r-1)!}$$

$$=\frac{1}{(2r+2)!}\cdot\frac{\{(n+1)(2r+2)\cdot(n)!-(2r+1)(n+1)!\}}{(n-2r-1)!}$$

$$=\frac{1}{\{2(r+1)\}!}\cdot\frac{(n+1)!}{(n-2r-1)!}$$

$$=\frac{1}{\{2(r+1)\}!}\cdot(n-2r)(n-2r+1)\cdot\cdots\cdots\cdot n\cdot(n+1)$$

となり，$r=r+1$に対しても式（付 2.1.1）は成立する。

故に式（付 2.1.1）は一般のrに対して成立することが証明された。

付録2.2　式（2.37）の導出

式（2.34）の右辺第1項を定数Cとおき，式（2.33）と（2.36）を用いると式（2.34）は以下のように変形される。

$$E \simeq C - \frac{1}{k \cdot T_0} \sum_{L_j \neq L_W}^{k \cdot m_0} f_H(L_j) \{ \log_2 f_H(L_j) - (\log_2 T_0 + \log_2 k) \}$$

$$= C - \frac{1}{k \cdot T_0} \{ \sum_{L_j \neq L_W}^{k \cdot m_0} f_H(L_j) \cdot \log_2 f_H(L_j) - (\log_2 T_0 + \log_2 k) \cdot \sum_{L_j \neq L_W}^{k \cdot m_0} f_H(L_j) \}$$

$$\simeq C - \frac{k}{k \cdot T_0} \{ \sum_{L_i \neq L_{W_0}}^{m_0} f_{H_0}(L_i) \cdot \log_2 f_{H_0}(L_i) - (\log_2 T_0 + \log_2 k) \sum_{L_i \neq L_{W_0}}^{m_0} f_{H_0}(L_i) \}$$

$$= C - \frac{1}{T_0} [\sum_{L_i \neq L_{W_0}}^{m_0} \{ f_{H_0}(L_i) (\log_2 f_{H_0}(L_i) - \log_2 T_0 \} - \log_2 k \sum_{L_i \neq L_{W_0}}^{m_0} f_{H_0}(L_i)]$$

$$= \{ C - \sum_{L_i \neq L_{W_0}}^{m_0} \frac{f_{H_0}(L_i)}{T_0} \cdot \log_2 \frac{f_{H_0}(L_i)}{T_0} \} + \frac{\log_2 k}{T_0} \cdot \sum_{L_i \neq L_{W_0}}^{m_0} f_{H_0}(L_i)$$

$$= E_0 + (1 - \frac{N_{V0}}{T_0}) \cdot \log_2 k$$

以上，式（2.37）が得られた。

ここで

$$E_0 = C - \sum_{L_i \neq L_{W_0}}^{m_0} \frac{f_{H_0}(L_i)}{T_0} \cdot \log_2 \frac{f_{H_0}(L_i)}{T_0}$$

である。

付録2.3　式（2.39）の導出

2値画像を対象に白，黒分離ランレングス符号化を考える。

白，黒画素の生成が単純マルコフ過程に従うものとして，白ラン，黒ランのエントロピー q_R^W，q_R^B をそれぞれ平均ランレングス V_W，V_B を用いて表わすと次のようになる。(59)

$$q_R^W(m)=V_W(m)\cdot h(V_W(m)) \qquad (付2.3.1)$$

$$q_R^B(m)=V_B(m)\cdot h(V_B(m)) \qquad (付2.3.2)$$

$$h(y)=-\frac{1}{y}\cdot\log_2\left(\frac{1}{y}\right)-\left(1-\frac{1}{y}\right)\cdot\log_2\left(1-\frac{1}{y}\right)$$

但し，$h(V_W)$，$h(V_W)$はそれぞれ白ラン，黒ランの平却値がV_W，V_Bとなるパターンでの白，黒画素当りのエントロピーである。

式（付2.3.1）と（付2.3.2）を用いて，サイズmの変化に対する各ランのエントロピーの変化を調べる。

サイズの変化に伴なうパターンの相似的な変化に対して，平均ランレングス長はサイズに比例して変化するので，

$$V_W(m)=k\cdot V_W(m_0) \qquad (付2.3.3)$$

$$V_B(m)=k\cdot V_B(m_0) \qquad (付2.3.4)$$

但し　$m=k\cdot m_0$

となる。

したがって

$$q_R^W(m)=k\cdot V_W(m_0)\cdot\left\{-\frac{1}{k\cdot V_W(m_0)}\log_2\frac{1}{k\cdot V_W(m_0)}\right.$$

$$\left.-\left(1-\frac{1}{k\cdot V_W(m_0)}\right)\cdot\log_2\left(1-\frac{1}{k\cdot V_W(m_0)}\right)\right\}$$

$$=\log_2\left(k\cdot V_W(m_0)\right)-\left(k\cdot V_W(m_0)-1\right)\log_2\left(1-\frac{1}{k\cdot V_W(m_0)}\right)$$

$$=\log_2\left(k\cdot V_W(m_0)\right)-\left(V_W(m_0)-1\right)\log_2\left(1-\frac{1}{V_W(m_0)}\right)$$

$$+\left(V_W(m_0)-1\right)\cdot\log_2\left(1-\frac{1}{V_W(m_0)}\right)$$

$$-\left(k\cdot V_W(m_0)-1\right)\log_2\left(1-\frac{1}{k\cdot V_W(m_0)}\right)$$

$$=V_W(m_0)\left\{-\frac{1}{V_W(m_0)}\cdot\log_2\frac{1}{V_W(m_0)}-\left(1-\frac{1}{V_W(m_0)}\right)\cdot\right.$$

$$\left.\log_2\left(1-\frac{1}{V_W(m_0)}\right)\right\}+\log_2 k+\Delta\left(k,V_W(m_0)\right)$$

$$=q_R^W(m_0)+\log_2 k+\Delta\left(k,V_W(m_0)\right) \qquad (付\ 2.3.5)$$

となる。

但し，

$$\Delta(k,v)=\log_2\frac{\left(1-\frac{1}{v}\right)^{v-1}}{\left(1-\frac{1}{kv}\right)^{kv-1}} \qquad (付\ 2.3.6)$$

$$(v\geqq 1,\ kv\geqq 1)$$

である。

同様に

$$q_R^B(m)=q_R^B(m_0)+\log_2 k+\Delta\left(k,V_B(m_0)\right) \qquad (付\ 2.3.7)$$

を得る。

付録2.4　サイズとデータ圧縮比の検討に使用した試料

円，五，物，勢，球，毛，階，価，桑，憲，紐，城，界，鮫，
喉，制，堀，台，羽，宰，臓，煙，蛾，派，髪，於，景，請，
覚，剰，裟，勧，内，零，卯，幽，仮，椿，敏，繋

第3章　画素形漢字パターンのサイズ変換処理

3.1　まえがき

代表的な電子的漢字パターン発生方式として画素方式があるが，発生される漢字パターンの大きさは漢字パターン発生器内にあらかじめ格納されている画素形漢字パターンで限定され，変更することは不可能であった。

一方，漢字パターンの利用面から考えると1枚の文書の作成においても，本文用，見い出し用，脚注用，さらに表，グラフなど図形中への挿入用など種々の大きさの漢字パターンが必要となる。(64)

さらに，また，編集用のディスプレイ装置と記録をとるためのハードコピー装置など解像度の異なる複数種類の出力装置を含む複合システムを考えると，要求される漢字パターンの種類は，さらに増えることになる。

しかし，漢字は字種が多く，かつ，字形が複雑であるため，大きさの異なる複数種類の漢字パターンを全て漢字パターン発生器内に格納すると，漢字パターン発生器が極めて高価なものとなってしまうため，上記の要求に応えることは現実的には不可能となっている。

そこで本章では，1つの基準の漢字パターンを基に，その構成画素数を変えて，必要な大きさの漢字パターンを作成する画素形漢字パターンのサイズ変換処理について考察する。なお，光学的な手法での表示サイズの変更と明確に区別する意味で，漢字パターンを1，0の要素から成るマトリクスと見なし，サイズ変換のことを，漢字パターンマトリクスの次数変換と呼ぶことも多い。(28)

画素形漢字パターンのサイズ変換法の研究としては，1975年に黒崎が発表した(4-3)梱包ラベル，荷札用の漢字パターンを得るための整数倍拡大補間処理の考察が見られるのみであり，非整数倍まで含めて拡大，縮小法に関する一般的なサイズ変換処理の研究は本研究が最初であると思われる。

その後，多くのサイズ変換処理に関する研究がなされているが，画素形漢字パターンのサイズ変換の考え方としては，漢字の特徴を利用する考え方と，一般的な2値図形としてサイズ変換を行ない，その後補正処理を行なう考え方とに分かれる。

本研究は前者の，漢字パターンの性質を積極的に考慮してサイズ変換を行なう考え方に

たつものであり，同じ考え方の研究として，中野，他[65]と渡部，他[30]，井面，他[66]の研究がある。

一方，一般の2値図形とみたサイズ変換処理としては，井上，他[33]，大川，他[34]，斉藤，他[35]，塩野，他[67]，小川，他[38]，渡辺，他[68]，清水，他[69]，の方法が報告されている。

以下，本章では，サイズ変換処理に関する用語，要求条件など問題点の抽出，周波数空間上での考察，画素の選択基準の設定法に対する基礎的考察，漢字パターンの特徴を利用した“線分の比例配置法”の考え方と実験結果，歪補正処理，サイズ変換処理の高速化手法，適用領域について述べる。

3.2 予備的考察

3.2.1 用語

画素形漢字パターンのサイズ変換処理について考察する前にいくつかの用語について定義する。

（m行×n列）の白，黒画素の配列として表現された画素形漢字パターン（以下，特にことわらない限り，画素形漢字パターンを単に漢字パターンとよぶ。）Cを

$$C=\{\ a(i,\ j)\},\quad i=1\sim m,\quad j=1\sim n \qquad (3.1)$$

$$a(i,\ j)=\begin{cases}0：白画素\\ 1.：黒画素\end{cases}$$

と表わす。すなわち，$a(i,\ j)$はi行，j列に位置する画素の状態を示している。以下式（3.1）で定義される漢字パターンの行数mを縦サイズ，列数nを横サイズ，また，併わせて単にサイズとよぶ。

縦サイズと横サイズがそれぞれm．nの漢字パターンを（$m\times n$）の漢字パターンと表記する。

与えられたサイズ（$m\times n$）の漢字パターンを基に，異なるサイズ（$M\times N$）の漢字パターンを作成する処理を以下“サイズ変換”処理とよぶ。

漢字パターンCはサイズ変換処理により（M行×N列）の画素配列から成る漢字パ

ターンC^Tに変換される。この時，変換前の漢字パターンCを原漢字パターン，変換後の漢字パターンC^Tを変換漢字パターンとよび，

$$R_i = \frac{M}{m}, \quad Rj = \frac{N}{n} \qquad (3.2)$$

をそれぞれ縦変換倍率，横変換倍率と定義する。特に，縦，横の区別が必要ないときは，単に変換倍率とよぶ。

変換倍率が1より小さい場合を縮小サイズ変換（あるいは縮小変換），変換倍率が1より大きい場合を拡大サイズ変換（あるいは拡大変換）とよぶ。また，拡大または縮小をサイズ変換の方向とよぶ。

3.2.2 サイズ変換に対する要求条件

漢字パターンのサイズを変換するためには，拡大変換に対しては画素の挿入が，縮小変換に対しては画素の削除を行なうことが必要である。このような画素の挿入，削除を不適切に行なえば，漢字パターンを構成するのに必要な字画が欠落して誤字となったり，あるいは，そうでなくとも，大きな歪を生じ文字品質を著るしく劣化させることになる。

個々の文字パターンの品質に関する研究は今まで，殆んどなされておらず，したがって，客観的な文字パターンの品質評価基準は現時点では確立されていない。

そこで，サイズ変換処理の考察を進める上で以下の条件を設定する。

〔条件1〕：変換漢字パターンが誤字とならないこと。

〔条件2〕：変換漢字パターンの品質劣化が小さいこと。

条件2は，具体的には次の3つの条件とする。

〔条件2－1〕：変換漢字パターンと原漢字パターンは出来るだけ相似変換に近いこと。

〔条件2－2〕：変換漢字パターンにおいて，縦直線部と横直線部の線幅はそれぞれ一定に統一されること。

〔条件2－3〕：変換漢字パターンにおいて，構成線分の周辺は出来るだけ滑らかであること。

上記の要求条件の中で〔条件1〕と〔条件2－1〕は変換の方向あるいは変換倍率の範囲によらず，等しく満足されるべき条件であるが，〔条件2－2〕は変換漢字パ

ターンのサイズが（３２×３２）程度以下の，比較的小サイズとなる場合に重要な条件となり，一方，〔条件２－３〕は変換漢字パターンが大サイズとなる場合に重要となる。

以上は変換漢字パターンに対する文字品質面からの要求条件であったが，他に，サイズ変換処理に要する時間，サイズ変換処理のための付加情報の使用の可否などの面からの要求条件があげられるが，これらの条件については3.5節のサイズ変換処理の適用領域についての考察として述べる。

3.2.3 周波数領域での考察

漢字パターンのサイズ変換は，与えられた漢字パターンをサンプリングし，所定のサンプル点から成る画像を得ることと考えられる。

この時，与えられた原画像を完全に復元するために必要なサンプリング密度は標本化定理によって求められる。

しかし，本章で扱かう漢字パターンのサイズ変換の問題は，逆にサンプリング密度が指定された時，如何に歪の少ないパターンを得るかということになる。

一般に，与えられた画像を粗い密度でサンプリングすれば，得られる周波数スペクトルは空間周波数領域上で図3.1に示すようになり，合同な図形群の１部が互いに重なり部分をもつことになる。その基底スペクトルを用いて画像を復元しても，いわゆる折り返し歪が生じて原画像の完全な復元は出来ないことは良く知られている。

このように，原画像がもつ周波数スペクトルの広がりに対して，粗い密度でサンプリングする場合には，サンプリングする前に，与えられた画像に対して前処理フィルタリングを行ない周波数帯域を制限することにより，折り返し歪を小さくすることが出来るとされている。(70)

そこで，本項では先ず，折り返し歪の無い多値の漢字パターンへとサイズ変換し，その後２値化する方法について考察する。

図3.2は実験に用いた（１２８×１２８）の原漢字パターンである。この原漢字パターンに対して離散的フーリエ変換を行ない，得られる周波数領域上のデータに対して，図3.3に示すフィルタ（図で斜線部は１，その他は０の通過性を持つ）をかけ，原漢字パターンの周波数帯域を制限する。その後，逆フーリエ変換を行ない，その絶対値を取ることにより，帯域制限を行なった（１２８×１２８）の多値漢字パターン

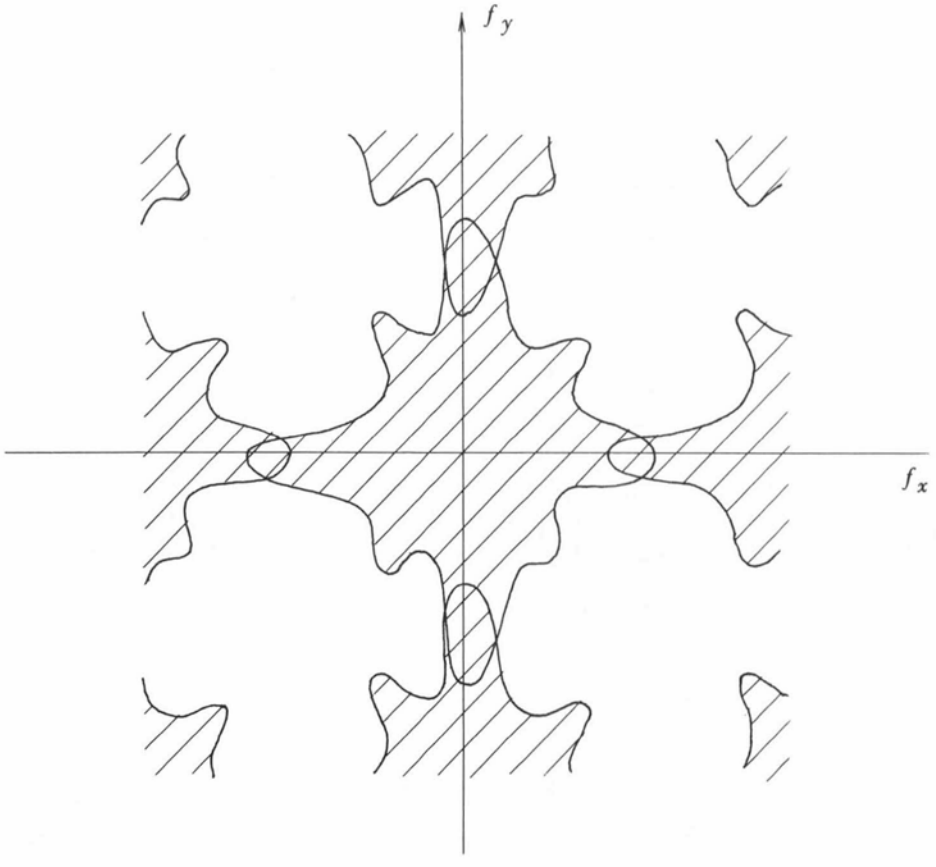


図 3.1　粗い密度でサンプリングされた画像のフーリェスペクトル

を得る。その結果を図 3.4 に示す。

さらに，この帯域制限された（１２８×１２８）の多値漢字パターンを縦，横方向に 4 画素おきにサンプリングして，（３２×３２）に縮小サイズ変換した多値漢字パターン図を 3.5 に示している。これは前処理フィルタリングにより原漢字パターンの帯域制限を行ない，折り返し歪を取り除いて得られる縮小サイズ変換処理の結果を与える。

（128×128）

図 3.2　原漢字パターン（２値）

最後に，この（３２×３２）の多値漢字パターンに対して閾値処理を行ない 2 値漢字パターンを得る。

すなわち，多値漢字パターン内の最大画素値 A_{max} を基準として，以下の閾値 A_{th} を設定し 2 値化した。

$$A_{th} / A_{max} = R \qquad (3.3)$$

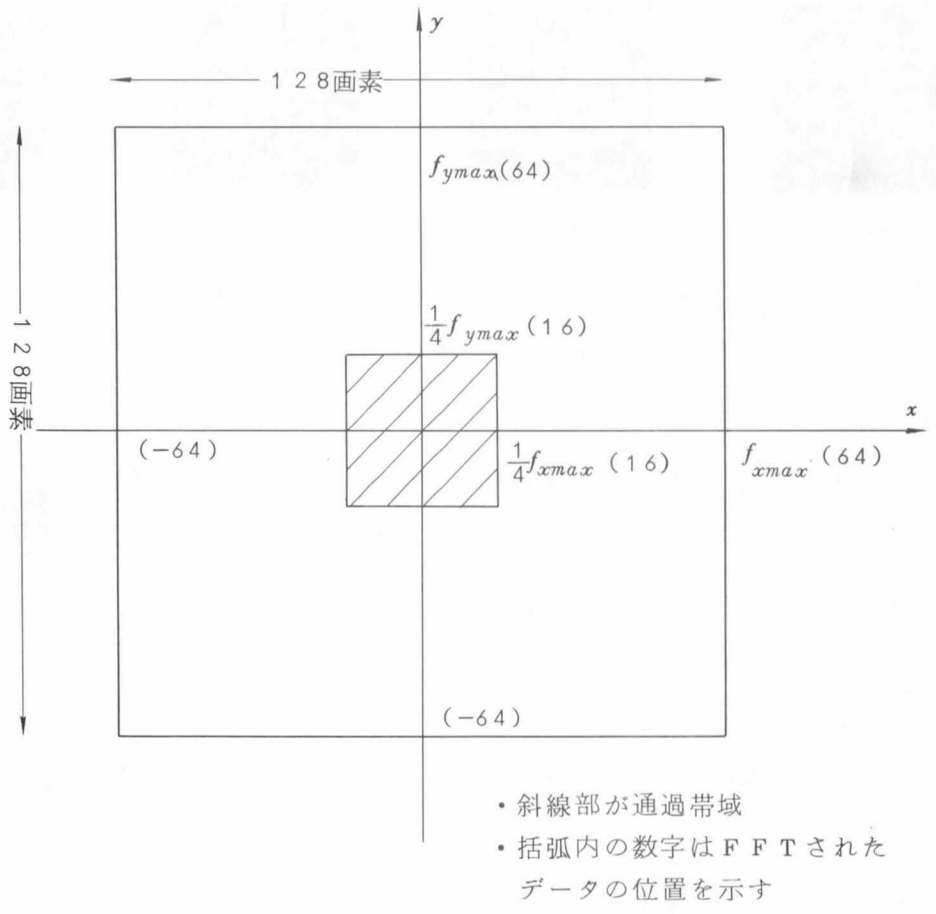
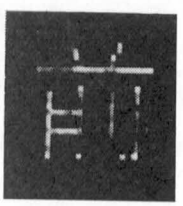


図 3. 3　帯域制限に使用したフィルタの形

図 3. 6 の(a)～(d)に，２値化した漢字パターンの例を示している。また，比較の対象として，図 3. 2 に示した（１２８×１２８）の２値の原漢字パターンを直接サンプリングして得た（３２×３２）の縮小漢字パターンの例を図 3. 7 に示している。

（128×128）

図3. 4　帯域制限後
多値漢字パターン

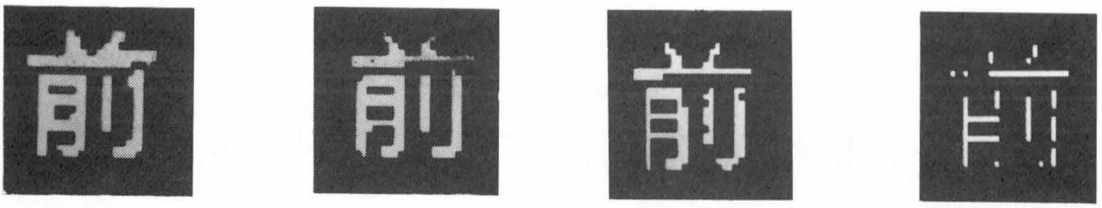


（32×32）

図3. 5　帯域制限後
サンプリングされた
漢字パターン
（多値）

図 3. 6 の(a)～(d)に示す例から分かるように，漢字パターンの品質は２値化により大きな量子化歪を受けて，劣化してしまい，図 3. 7 の直接サンプリングして得られる結果と比較しても，折り返し歪を除去した効果は現われないと云える。

すなわち，サイズが（３２×３２）の漢字パターンでは漢字の構成線分の幅は２，



(a) R＝0.2　(b) R＝0.4　(c) R＝0.6　(d) R＝0.8

図3.6　帯域制限後の多値（128×128）漢字パターンを（4×4）のサブブロックを単位として閾値処理した（32×32）の2値パターン

3画素であり，2値化に伴ない発生する線幅の1画素の不揃いが文字品質劣化の支配要因となり，折り返し歪除去の効果が量子化歪に消されてしまうことが分かる。

以上の実験結果を基に，本研究の主たる対象であるサイズ（16×16）～（64×64）の範囲の2値漢字パターンのサイズ変換処理では，周波数空間上での処理は処理量の増大の割に，文字品質上の改善効果は少ないと判断して，実空間上の処理に限定して行なうこととした。

（32×32）

図3.7　（128×128）の原漢字パターンを直接サンプリングした（32×32）2値漢字パターン

3.2.4　比例法 (71)

最も直感的な漢字パターンのサイズ変換法として，原漢字パターンの画素$a(i, j)$を次式で与えられる変換漢字パターンの画素$a^t(I, J)$に対応させる比例的な写像変換が考えられる。

$$\left.\begin{aligned} & a^t(I, J) = a(i, j) \\ & \text{但し,} \\ & I = [\, i \times R_i + 0.5 \,] \\ & J = [\, j \times R_j + 0.5 \,] \end{aligned}\right\} \qquad (3.4)$$

ここで，R_i，R_jは式（3，2）で定義される変換倍率，〔 〕はガウス記号である。

この単純な写像変換では，拡大サイズ変換時には変換漢字パターン上に値の不定な画素が生じるし，縮小サイズ変換時には変換漢字パターンの画素が2つの値をとる場合が生じる。

すなわち，数学的には，式（3.4）は拡大サイズ変換に対しては全射（Surjec-

tion)でなく，また，縮小サイズ変換に対しては単射（injection）ではない。

(1) 比例法

そこで，次の写像を定義することによって変換漢字パターンを一意的に得ることにする。

サイズ変換に伴なう字画の欠落を避けるために画素の状態に優劣を付け，黒画素状態が白画素状態に優先するものとする。この結果，以下の変換処理は原漢字パターンの黒画素に対してのみ実行すれば良く，変換処理の高速化が図れる。

(i) 縮小サイズ変換の場合

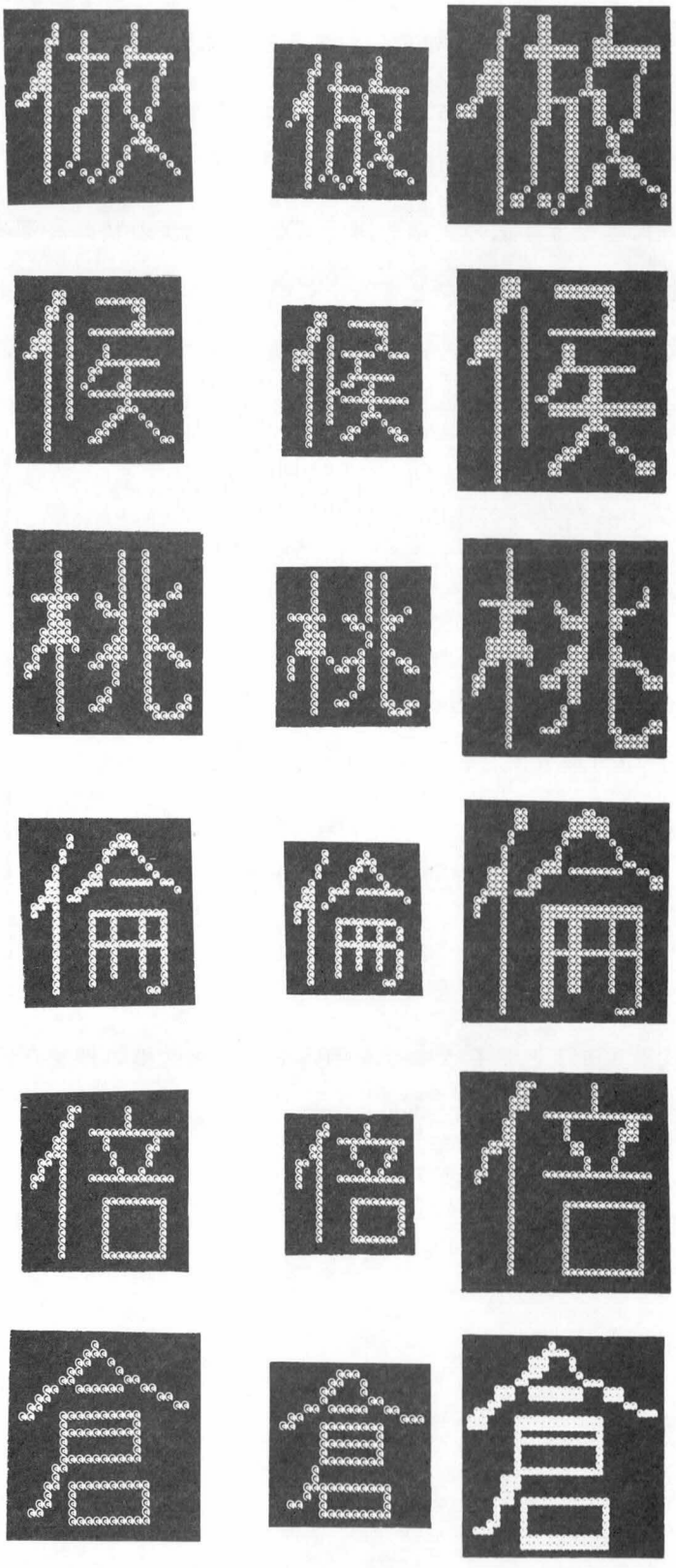
$$\left.\begin{aligned} & a^t(I, J) = a(i, j) \quad \text{for } a(i, j) = 1 \\ & I = [\, i \times R_i + 0.5 \,] \\ & J = [\, j \times R_j + 0.5 \,] \\ & i = 1, 2, \cdots, m, \qquad j = 1, 2, \cdots, n \end{aligned}\right\} \qquad (3.5)$$

(ii) 拡大サイズ変換の場合

$$\left.\begin{aligned} & a^t(I, J) = a(i, j) \quad \text{for } a(i, j) = 1 \\ & [\, i \times R_i + 0.5 \,] \leqq I < [\, (i+1) \times R_i + 0.5 \,] \\ & [\, j \times R_j + 0.5 \,] \leqq J < [\, (j+1) \times R_j + 0.5 \,] \\ & i = 1, 2, \cdots, m \\ & j = 1, 2, \cdots, n \end{aligned}\right\} \qquad (3.6)$$

以下，式（3.5）と式（3.6）を用いたサイズ変換処理を“比例法”とよぶことにする。

図3.8に比例法による変換漢字パターンの例を示す。



(a)原漢字パターン
（24×24）

(b)縮小変換漢字
パターン
（20×20）

(c)拡大変換漢字
パターン
（32×32）

図 3.8　比例法による変換漢字パターン例

(2) 比例法に対する考察

3.2.2項で述べたサイズ変換処理に対する要求条件に照らし合わせて比例法を見るとき，次のことが指摘される。

① 基本的に漢字パターンの相似変換を実現するものであり，文字バランスの劣化は無い。

② 縦，横直線の線幅に不揃いが生じる。

③ 画素状態において黒画素状態を優先させるため，縮小サイズ変換では線分間隙が消滅する場合があり，変換漢字パターンが誤字となることがある。

比例法における上記の②，③の欠点は，比例法が画素の挿入，削除を，個々の漢字パターンの形状を無視して，全ての漢字パターンに対して一律に，同一ケ所に行なうために生じるものである。

この欠点は，同じ個数の画素の挿入，削除を行なうにしても，漢字パターン対応にその挿入，削除位置を適切に選択することによって解決することが出来る。

3.2.5 行列選択法 (72)

(1) 行および列を単位とした画素の挿入，削除

画素形漢字パターンのサイズ変換を行なうためには画素の挿入あるいは削除が必要である。したがって，サイズ変換処理の問題は挿入あるいは削除を行なう画素位置の選択の問題ということができる。

本研究においては，

① 漢字パターンは多数の縦，横直線を含むこと，

② 縦，横直線の直線性を保存すること，

③ 縦，横直線の周辺凹凸（以後，ノッチと呼ぶ）を生じさせないこと，

の条件から，画素の挿入，削除は漢字パターンの行および列を単位として行なうこととする。

(2) 行列選択法

画素の挿入，削除を行および列を単位として行なうとき，サイズ変換処理の問題は挿入，削除を行なう行および列の選択の問題となる。

前述の比例法も結果的には画素の選択を行，列を単位として行なったことになるが，以下，考察する行列選択法は，個々の漢字パターン対応に挿入，削除する行（および

列）を選択する点で比例法と基本的に異なる。このように，漢字パターン対応に行（および列）を選択する考え方は従来にない新しいものであり，このことにより，縮小変換まで含めた非整数倍のサイズ変換の可能性が出てくる。

なお，以下，行および列の挿入とは，所定の走査方向で見て直前の行および列を反復追加することを意味し，削除とは，その行および列を直前の行および列に重畳し，それ以降の行および列を平行移動することを意味する。

(i) 選択基準

以下の行および列の選択基準を設ける。

〔基準１〕：漢字パターンを構成する縦直線を含む列および横直線を含む行は挿入あるいは削除しない。但し，縦，横直線の線幅の変更を伴う場合には，縦直線を含む列および横直線を含む行は，それぞれ同時に挿入，削除を行なう。

〔基準２〕：その行および列を挿入，削除することによって生じる視覚的な漢字パターンの差異が最も少ないものから順次挿入，削除する。

〔基準２〕はさらに次のように具体化する。

すなわち，「類似した画素配列をもつ行または列が多数連続する部分から削除したり，あるいはその部分へ類似した行または列を挿入する場合が，漢字パターンの変化から受ける視覚的影響は少ない」と考えることにより，次の〔基準2'〕を得る。

〔基準2'〕：隣接する行と画素配列の類似度の高い行，隣接する列と画素配列の類似度の高い列から順次，挿入あるいは削除を行なう。

(ii) 行，列の重みづけ

前述の基準に基づく行，列の選択を実現するため次の３つの分布を導入する。

(a) 周辺分布：$X(j)$，$Y(i)$

$$\left.\begin{aligned} X(j)&=\sum_{i=1}^{m} a(i,j),\quad j=1,2,\cdots,n \\ Y(i)&=\sum_{j=1}^{n} a(i,j),\quad i=1,2,\cdots,m \end{aligned}\right\} \qquad (3.7)$$

$X(j)$，$Y(i)$をそれぞれ横周辺分布，縦周辺分布とよぶ。(73)

(b) 変化画素分布：$DX(j)$，$DY(i)$

$$DX(j)=\sum_{i=1}^{m}\left(a(i,\ j-1)\oplus a(i,\ j)\right),\quad j=1,\ 2,\ \cdots,\ n$$

$$DY(i)=\sum_{j=1}^{n}\left(a(i-1,\ j)\oplus a(i,\ j)\right),\quad i=1,\ 2,\ \cdots,\ m \qquad (3.8)$$

ここで$a(0,\ j)=a(i,\ 0)=0$とし，⊕は排他的論理和を示す。

(c) 最大連続黒画素分布：MX(j)，MY(i)

$$MX(j)=max\{L(k,\ j)\}$$

$$MY(i)=max\{L(i,\ k)\} \qquad (3.9)$$

ここでL(k，j)はj列内でk行を始点とする縦方向への黒画素の連続数，L(i，k)はi行内でk列を始点とする横方向への黒画素の連続数，$max\{L\}$は集合{ L }の最大値を示す。

図3.9に，これら3つの分布の例を示す。

上記の分布の定義から，横直線の大多数は縦周辺分布Y(i)の極大点を与える行内に存在し，同じく縦直線の大多数は横周辺分布X(j)の極大点を与える列内に含まれることになる。したがって，逆に，縦，横の周辺分布の極大点を検出することによって漢字パターンを構成する縦，横直線を含む行および列の殆んどを抽出することができる。

また，変化画素分布で分布値が小さな部分では類似した画素配列をもつ行または列が続いていると解釈することができる。

このように，周辺分布値と変化画素分布値を行および列の選択のための“重み”づけの要素として用いることができる。

しかし，周辺分布はその極大点と縦，横直線の対応は比較的確度高く成り立つが，分布値の小さな部分が，行および列の選択に際して“重み”としてもつ意味が不明確である。すなわち図3.10に示す「上」の漢字パターンの列についてみると，番号5の列以外の2～7列は挿入，削除の重みとしては全て等しい値をとるべきであるが，周辺分布値では明らかに番号6，7の列が番号2～4の列より大きな重みをもつことになる。

すなわち，横周辺分布についていえば，横直線と1回交叉するたびに，その列の周辺分布値は横直線の線幅の画素数だけ加算されるため，縦直線を含まないにもかかわらず，多数の横直線と交叉する列の周辺分布値は増大し，真に縦直線を

含む列が埋まって検出できない場合や，上に述べたように，本来，同等に扱うべき列のあいだで分布値に差が生じるという不都合が生じる。

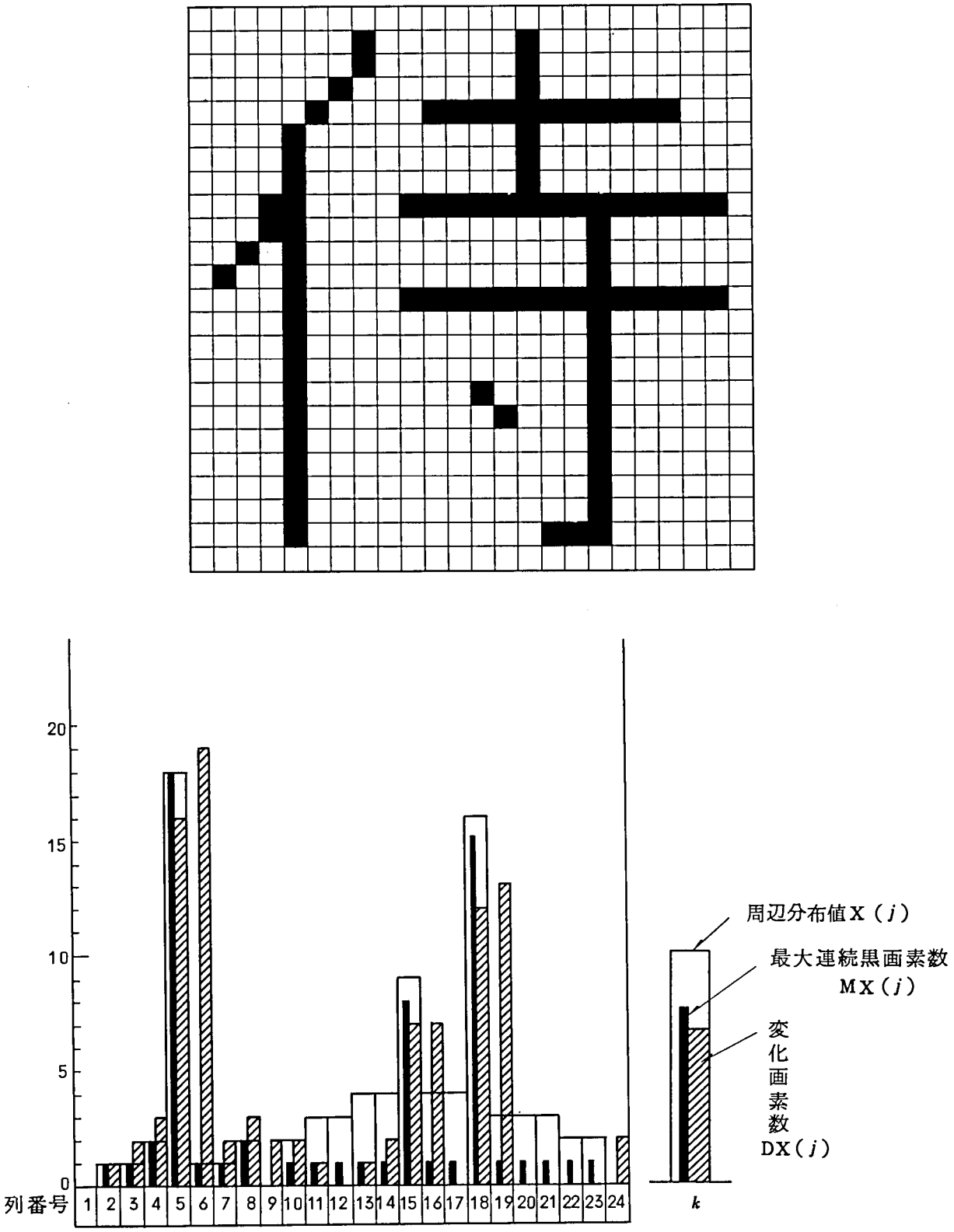


図 3.9　周辺分布，変化画素分布，最大連続黒画素分布の例

したがって，周辺分布は極大点などの分布値の大きな部分に対しては有効に“重み”分布として利用できるが，選択の〔基準１〕が述べるように，縦，横直線を含まない行および列から順次選択する場合に問題となる分布値の小さな行および列の重みづけには有効に利用できないという欠点がある。

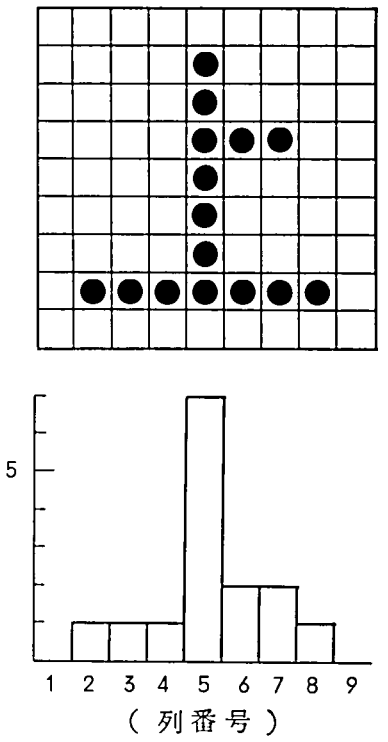


図3.10　「上」の横周辺分布

最大連続黒画素分布はこのような周辺分布がもつ不都合さを除去して，行および列の選択のために適した分布値を与え得る。

そこで，変化画素分布と最大連続黒画素分布を組み合わせることによって，行および列の“重み”を次のように定義する。

① 〔番号iの行の重みWR(i)〕

$$WR(i)=C_1\cdot DY(i)+C_2\cdot MY(i) \qquad (3.10)$$

② 〔番号jの列の重みWC(j)〕

$$WC(j)=C_1\cdot DX(j)+C_2\cdot MX(j) \qquad (3.11)$$

但し，DY(i)，MY(i)，DX(j)，MX(j)はそれぞれ式(3.8)，式(3.9)で定義される。

ここで，式(3.10)と式(3.11)で変化画素分布と最大連続黒画素分布の係数C_2とC_2について考察する。

(a) $C_1=0$の場合

すなわち，最大連続黒画素分布のみによる重み付けを行なう場合である。

この場合には，縦横直線を含まない行，列，すなわち，線分間隙部に相当する行，列のみが挿入，削除の対象となり，縮小変換時には線分間隙が消滅し誤字となる可能性が生じ，また，拡大変換時には縦横直線幅は一定であるため文字サイズの割には線幅が細い変換漢字パターンが得られることになる。

(b)　$C_2 = 0$ の場合

この場合には変化画素分布のみによる重み付けがなされることになるが，各縦横直線および線分間隙を構成する行，列の内，最低1個は大きな重み付けがなされるため，$C_1 = 0$ の場合に比較すると線分間隙の消滅により誤字となる可能性は小さくなる。

但し，縮小変換時に2画素以上の線幅の縦横直線が細められ，比較的変化倍率が小さな場合でも線幅の不揃いを生じる可能性が高い。

以上，定性的ではあるが，係数 C_1 は線分間隙を保持するために，係数 C_2 は縦横直線の線幅の不揃いを抑えるために有意であることが分かる。

C_1 と C_2 の最適比率は字種，漢字パターンのサイズ，書体等に依存して変化するものと考えられるが，以下，$C_1 = C_2$ として検討を進める。

この時，式（3.10），（3.11）で定義される行および列の重みは以下のようになる。

$$WR(i) = DY(i) + MY(i) \qquad (3.12)$$

$$WC(j) = DX(j) + MX(j) \qquad (3.13)$$

(iii)　行列選択法

行および列について順不同で走査し，式（4.12），（4.13）に定義した重みが小さい行，列から順次必要な個数だけ，縮小変換の場合は削除し，拡大変換の場合は挿入することにより，漢字パターンのサイズを変換する。

以後，このサイズ変換法を“行列選択法”とよぶ。

処理の流れを図3.11に示す。

また，この行列選択法を用いてサイズ変換を行なった例を図3.12に示す。同図で(a)はサイズ（24×24）の原漢字パターン，(b)はサイズ（20×20）の縮小変換漢字パターン，(c)はサイズ（32×32）の拡大変換漢字パターンの例である。

図3.12の実験例から次のことがいえる。

①　縦，横の直線幅の統一は比較的良好に実現される。

このことは，逆に，原漢字パターンの縦，横直線の線幅が1画素であれば，拡大次数変換において，拡大倍率がいかに大きくなろうとも縦，横直線の線幅は1画素のままであり，増加することはない。

②　字種によって著るしい文字バランスの劣化を生じる。

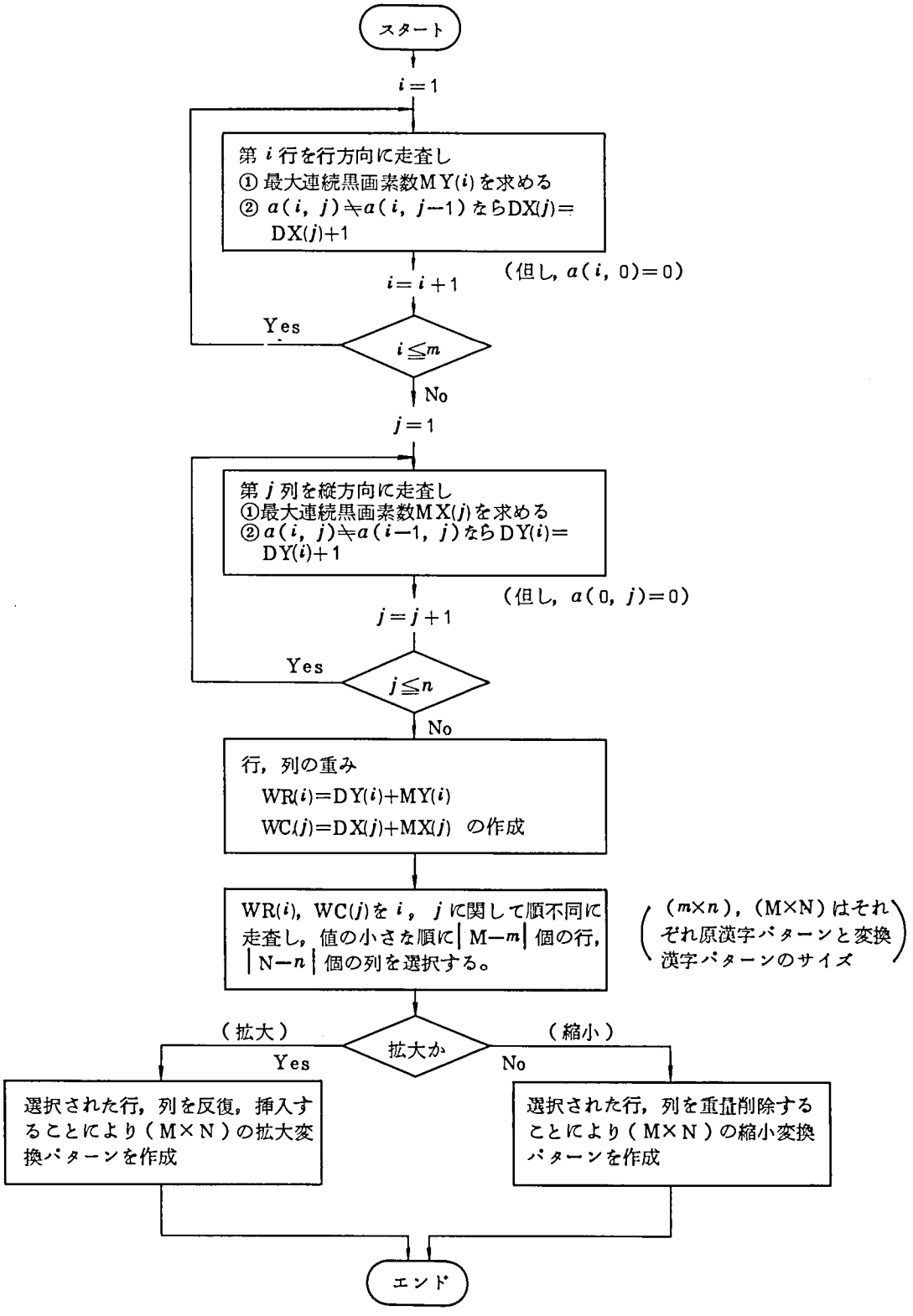
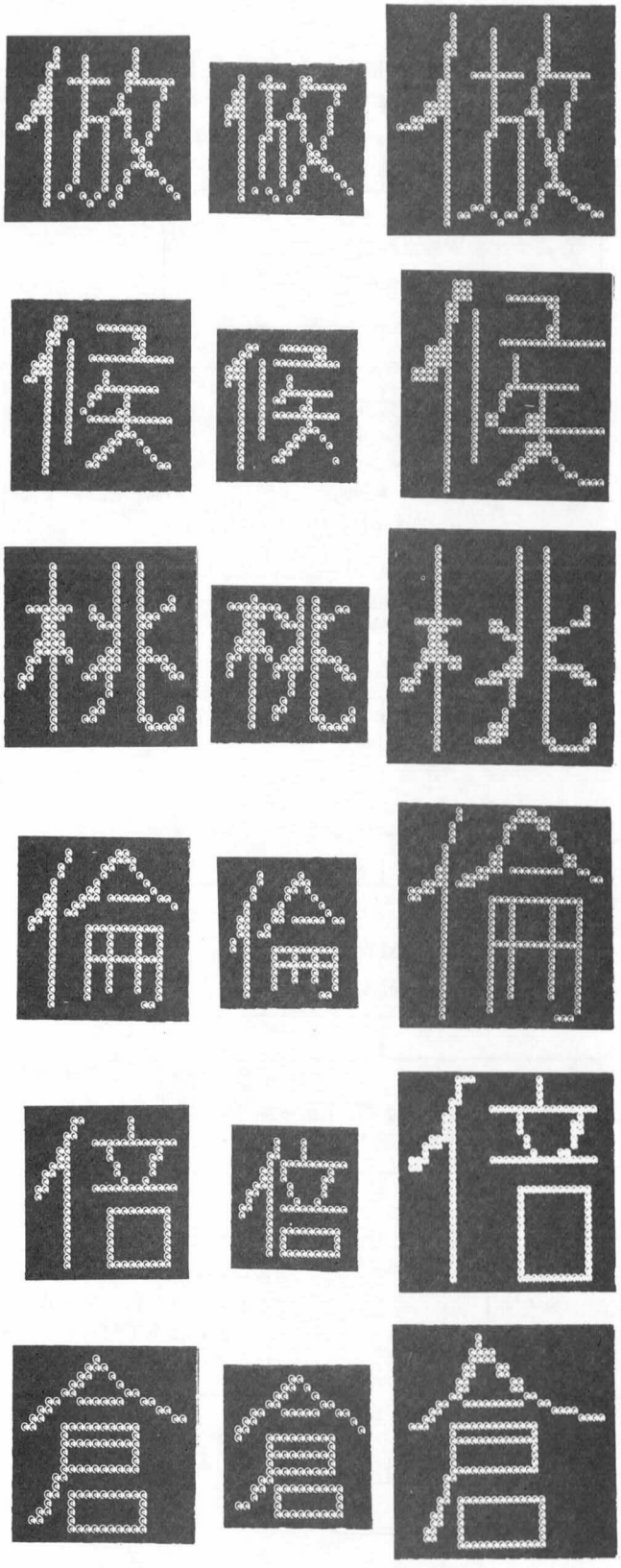


図 3.11　行列選択法の処理の流れ



(a)原漢字パターン (24×24)　(b)縮小変換漢字パターン (20×20)　(c)拡大変換漢字パターン (32×32)

図 3.12　行列選択法による変換漢字パターンの例

図3.12の例では，「倣」，「候」などのように線分密度が漢字パターンの全体にわたって比較的均一に分布している文字については文字バランスは保存されるが，「桃」，「倫」，「倍」，「倉」，などのように，サイズの変換方向に直交する直線の分布密度に偏りがある文字に対しては，分布の密度が小さな部分のみが部分的に拡大，縮小され文字バランスは著るしく劣化する。

(IV) 行列選択法における文字バランスの改善（領域分割形行列選択法）

行列選択法は基本的に漢字パターンを構成する縦横直線と線分間隙の保存を第1とした局所的な判断処理に基づくサイズ変換処理である。

局所的な判断処理に基づくため，行，列の選択のための重みに1でも差があると，重みの小さな行あるいは列が全て選択された後，初めて，その次に小さな重みをもつ行あるいは列が選択されることになる。したがって，明朝体漢字パターンのように，線端に“ウロコ”などの飾りをもつパターンでは文字バランスの劣化は更に大きくなる傾向となる。

サイズ（32×32）程度の漢字パターンにおいては行列選択法での重みの1～3程度の差は，上記の線端飾りや斜線，曲線部において生じる差が大部分を占めており，重みの小さな部分でのこの程度の差の無視により縦横直線の欠落が生じる可能性は小さい。

そこで，挿入，削除の対象となる比較的重みの絶対値が小さな部分における数画素分の重みの差の有意性は小さいと考え，文字バランスの改善法として，次の領域分割形行列選択法について考察する。

〔領域分割形行列選択法〕

「漢字パターンを上下および左右に等分割し，各分割区間内で選択される行あるいは列の個数の差が所定値K以下であるように，各区間毎に行列選択法を適用する」

すなわち，前述の行列選択法にしたがって，重みの小さな行および列から挿入，削除する行，列を選択してゆくが，その時，各区間で選択された行，列の個数を区間毎に登算しておき，その登算値の差が定数Kより大きくなった場合には，選択された行，列の個数が一番少ない区間については他の区間よりも1だけ大きな重みをもつ行，列までも選択の対象とすることにより，文字バランスの劣化を抑える方法である。

図3.13に「術」，「霜」，「汐」，「趣」の（32×32）の明朝体漢字パタ

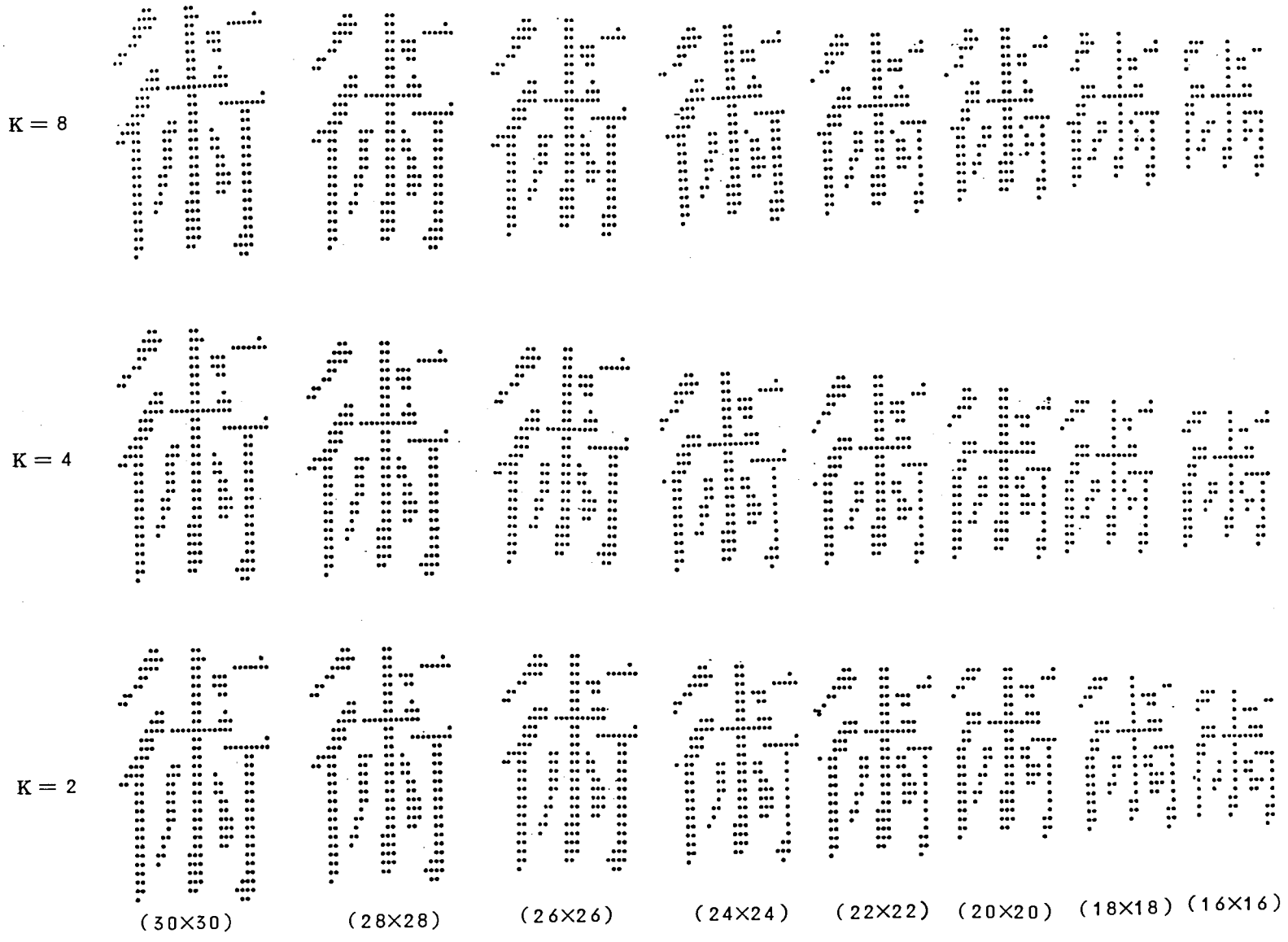


図3.13－1 領域分割形行列選択法による「術」の変換パターン

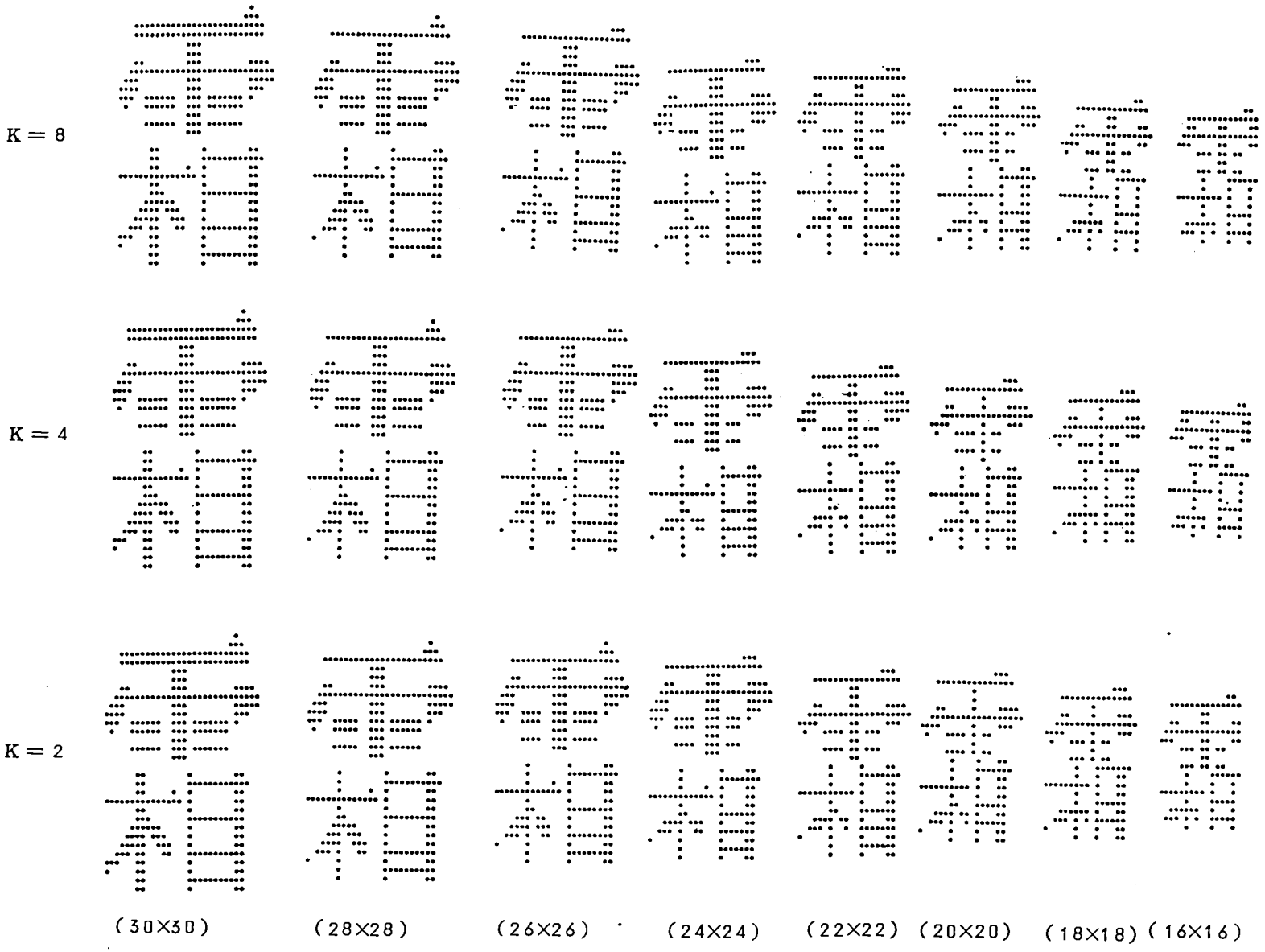


図3.13－2　領域分割形行列選択法による「霜」の変換パターン

K = 8
K = 4
K = 2
(30×30)
(28×28)
(26×26)
(24×24)
(22×22)
(20×20)
(18×18)
(16×16)

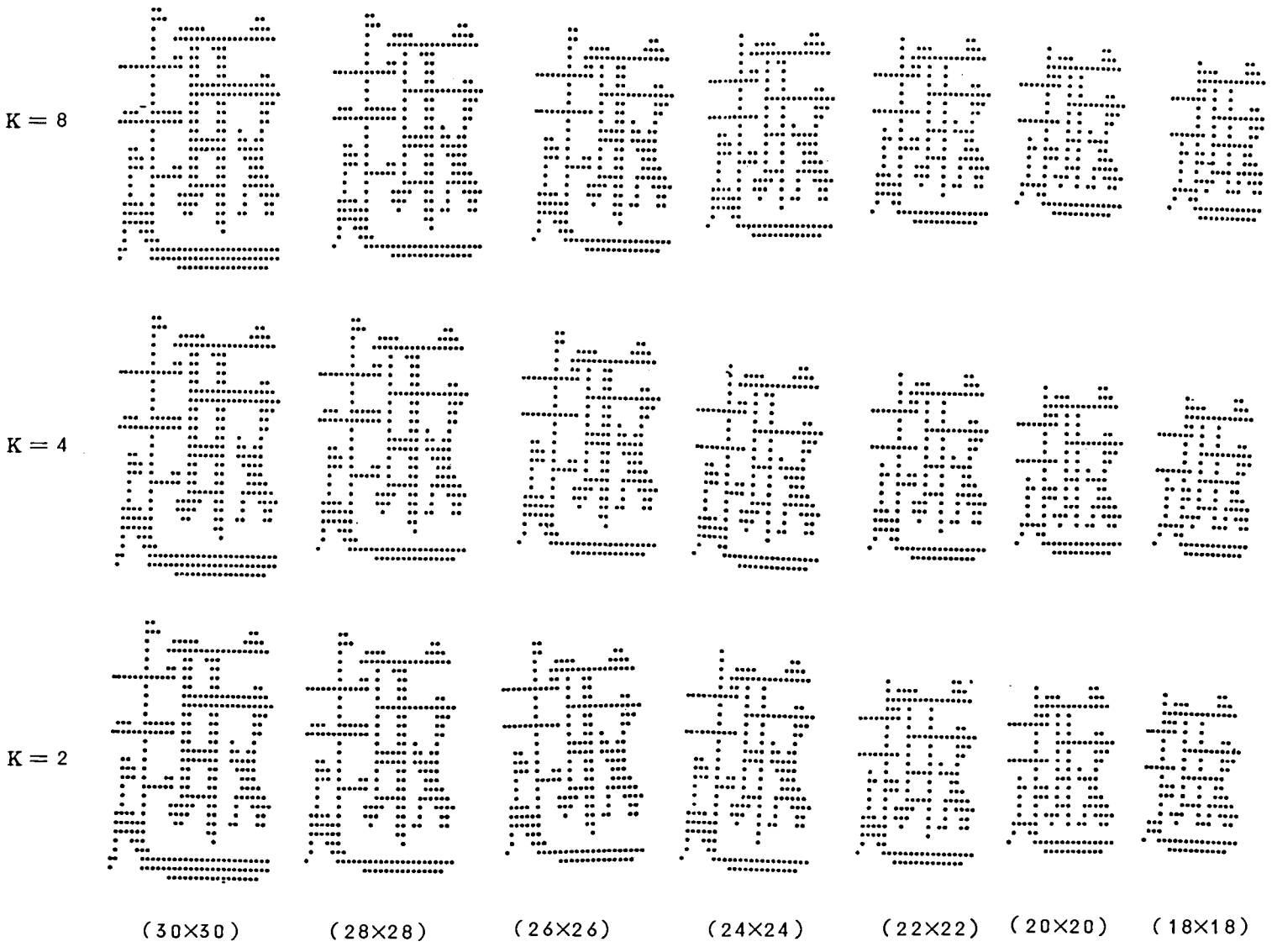


図 3.1 3 － 4　領域分割形行列選択法による「趣」の変換パターン

ーンを例にとって，K＝８，４，２とした時の縮小サイズ変換の結果を示している。

なお，この処理例では，文字バランスの改善効果の評価を目的とし，線幅の統一の条件を緩めて，選択のための重みとしては式（３.１０），（３.１１）でC_2＝０とした値を用いた。

この結果，これらの漢字パターンに対しては縮小サイズ変換は（２０×２０）～（２２×２２）が限界であり，この範囲内で定数Kの効果を見ると次の通りである。

① 縦横直線の分布に偏りがある，漢字パターン「術」に対しては，縦サイズ変換で見られるように比較的小さな変換倍率でも文字バランスの劣化を抑える効果は顕著である。

② 縦横直線の分布がほぼ一様である「霜」，「趣」や，斜線，曲線から成る「改」に対しては有意な差は生じない。

(V) 行列選択法の限界

行列選択法は漢字パターンを構成する縦横直線に対して大きな重みを与えるため，縦横直線はサイズ変換において最優先で状態が保存されることになる。

このことは同時に，拡大サイズ変換において漢字パターンが全体的に拡大してもそれに見合う線幅の拡大が行なわれないことになる。

したがって，行列選択法は縮小サイズ変換用の変換手法であるといえる。

そこで，文字バランスの劣化を抑えた領域分割形行列選択法を対象に主観評価実験を行ない縮小変換の範囲を調べた。

３０人の被験者を対象に，ファクシミリ受信機上に記録した，K＝２で得られた変換漢字パターンについて式（３.１４），（３.１５）で定義した了解度と明瞭度を求めた。

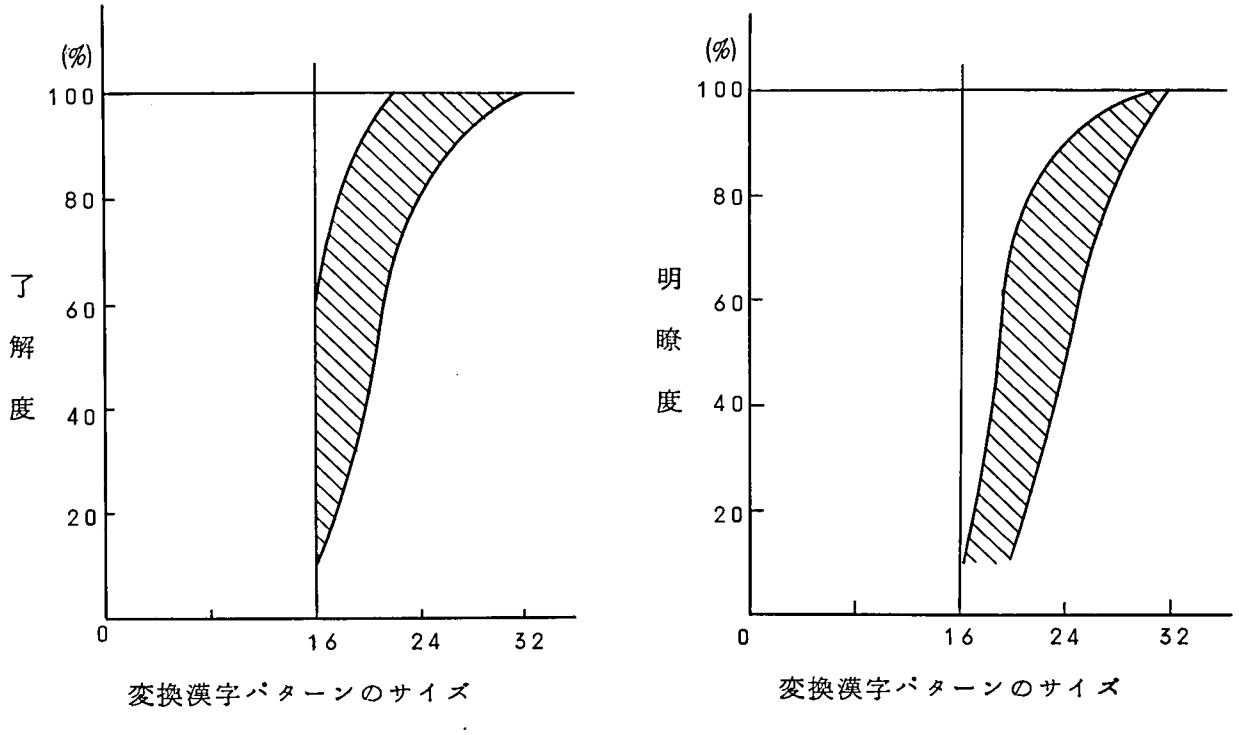
試料は「あ」，「人」，「右」，「応」，「初」，「受」，「奮」，「複」，「階」，「疑」，「燈」，「穀」，「熟」，「警」の１５文字であり，ファクシミリ受信機に１画素１ｍｍ方形で記録したものを写真を用いて縮小したものである。

図３.１４に評価実験で用いた試料，図３.１５に得られた了解度と明瞭度を示す。

なお，了解度とは誤りなく文字を判読できる度合であり，明瞭度とは，品質的に許容できる度合を意味しており，それぞれ以下の式で算出した。

$$了解度＝\frac{文字を読めた被験者数}{全被験者数（＝３０）}\times１００\,\% \qquad （３.１４）$$

図 3. 1 4　評価実験で用いた試料



(a) 変換漢字パターンのサイズと了解度　　(b) 変換漢字パターンのサイズと明瞭度

図3.15　変換漢字パターンに対する主観評価実験結果

$$\text{明瞭度} = \frac{\text{品質を許容できるとした被験者数}}{\text{全被験者数}(=30)} \times 100\,\% \qquad (3.15)$$

この結果，サイズ変換に伴う文字品質の劣化は激しいが，判読可能な範囲で考えるとかなり広い範囲で縮小サイズ変換が可能であることがわかる。すなわち，×24）程度までなら縮小された変換漢字パターンが，ディスプレイ用の文字パターンとして使用し得るものと考えられる。

3.3　線分の比例配置法 [28]

3.3.1　予備的考察

3.2節で述べた比例法と行列選択法の特徴をまとめると次のようになる。

① 縦，横直線の線幅の統一について

行列選択法では原漢字パターンの線幅が比較的良好に保存されるための線幅は統一

されやすいが，比例法では線幅の不揃いが生じる。

② 文字バランスの保持について

行列選択法では字種によって異なり，縦，横直線がパターン全体にわたって均一に分布している文字については文字バランスは保持されるが，縦，横直線の分布に偏りがある文字に対しては文字バランスの著しい劣化が生じる。他方，比例法は，当然のことながら，文字バランスは良好に保持される。

上記のように比例法と行列選択法は互いに相補的な特徴をもっていることが分かる。これらの特徴は，行列選択法では行および列の選択の重みづけを，局所的な判断で行なっていることから生じ，一方，比例法では個々の漢字パターンの構造を全く考慮していないことから生じている。

さらに，また，両者の違いは次のようにも考えられる。

行列選択法は与えられた漢字パターンを基に出来るだけ歪みが少ないように，かつ，誤字とならないように，漸次変形してゆくという図形の変形の発想であるのに対して，比例法は，字形を構成する黒画素を白の無地の上に順次はめ込んで新たに字形を構成する発想ともいえる。

すなわち，行列選択法は字形を構成する上で重要でない画素に注目した変形法であるのに対して，比例法は黒画素を白画素より優先させることにより，字形を構成する上で重要な画素に注目した方法である。

そこで，この比例法の考え方を発展させて，行，列に漢字パターンを構成する上での重要さに順位を付けて，重要な行および列から順次白無地上に比例的に配置して新たに漢字パターンを作成する考え方を線分の比例配置法ではとることとした。したがって，線分の比例配置法では，“重み”の大きな行，列を考察の対象とすることになり，文字バランスの保存と線幅の統一という，文字品質を左右する主要な要因の制御が可能となる。

しかし，単純に重みの大きな行，列を比例変換するだけでは画素単位の比例法と同じく縮小時の線分間隙の消滅と拡大時の線幅の不揃いは避けられない。

そこで，線分間隙の保存，線分幅の統一のための局所的な処理と，文字バランスを保存するための大局的な処理の2段階の処理ステップを踏むこととした。

3. 3. 2　線分の比例配置法

(1)　方法

原漢字パターン内であらかじめ縦横直線を含む行および列を抽出し，抽出された行および列を変換漢字パターン内に比例的に配置する。

具体的には線幅制御の必要な縦横直線と，縮小サイズ変換時に積極的に保存しなければ消滅する狭い線分間隙，すなわち，局所的な処理を必要とする部分（以後，線分部とよぶ）を抽出する。次に，線分部について線幅と線分間隙に対する変換処理を各線分部内で閉じて行ない，その変換後の線分部を漢字パターン内に比例的に配置する。

以後，上記の方法を“線分の比例配置法”とよぶ。

(2)　変換パラメータ

線分の比例配置法では表3. 1に示す各パラメータの制御が可能となる。

同表で最小線分間隙とは局所的処理の対象とする線分間隙である。通常，原漢字パターンの最小線分間隙となる。

表3. 1　変換パラメータ

パラメータ ＼ パターン		原漢字パターン		変換漢字パターン	
サイズ	横	n		N	
	縦	m		M	
線分幅	横	w_x	w	W_x	W
	縦	w_y		W_y	
最小線分間げき	横	d_x	d	D_x	D
	縦	d_y		D_y	

(3)　写像関数

原漢字パターンと変換漢字パターンのあいだでの行および列の対応づけを図3. 16を用いて説明する。

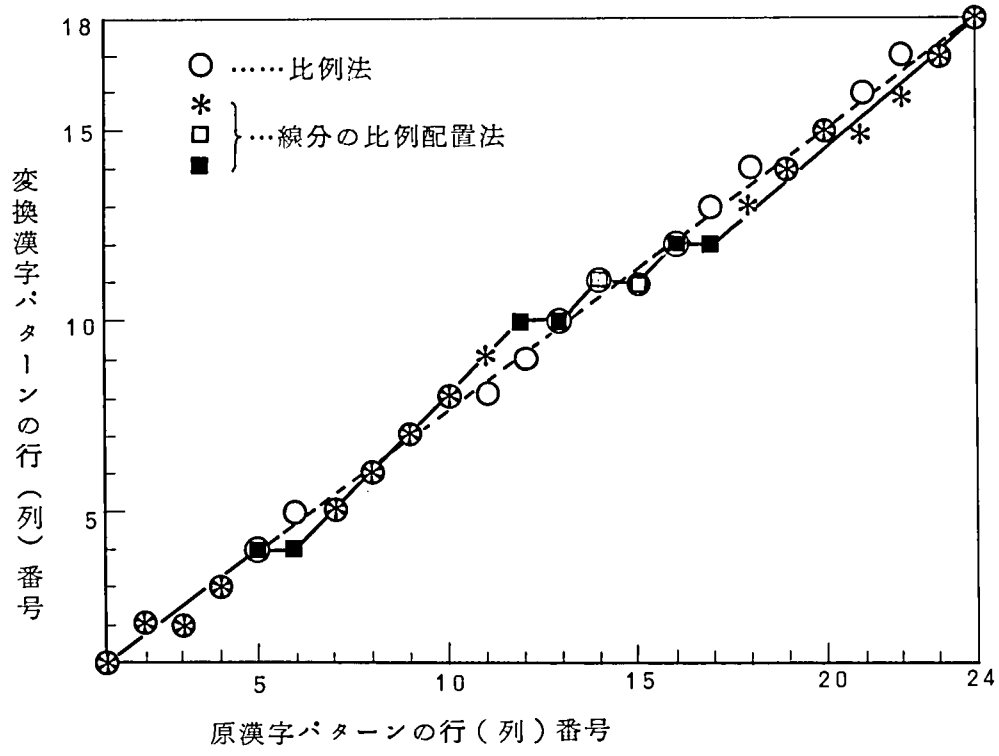


図3.16　原漢字パターンと変換漢字パターンの行（列）番号の対応関係

なお，以下の説明は行に対してのみおこなうが列に対する処理も全く同じである。

図3.16で横軸は原漢字パターンの行番号，縦軸は変換漢字パターンの行番号である。この例では縦サイズ24の原漢字パターンを縦サイズ18の変換漢字パターンへと縮小変換する場合を示しており，例えば，原漢字パターンの第8番行は変換漢字パターンの第6番行へと対応づけられる。

なお，表3.1に示した変換パラメータは$m=24$，$M=18$，$w=2$，$W=1$，$d=2$，$D=1$となる。

同図で破線に沿った○印は式（3.5）で与えられる比例法による対応関係を示し，実線で結ばれた□，■，＊の各印は線分の比例配置法による対応関係を示している。

原漢字パターンでは番号5，6，12，13，17の行位置に幅2ドットの線分が存在し，番号14，15の行位置に幅2ドットの最小線分間隙が存在している。なお，以後，最小線分間隙とそれをはさむ線分を含む部分を線分部とよぶ。この例では行番号12～17の範囲が線分部となる。

なお，上述の変換漢字パターンの線幅および最小線分間隙は変換漢字パターンのサイズに応じて適当な値が与えられるものとする。

図3.16では原漢字パターン内の3本の直線は■印の対応関係により線幅を1ドットに変換されて比例法による対応位置にできるだけ近く，それぞれ行番号4，10，12の位置に配置される。最小線分間隙も同じく□印の対応関係により変換漢字パターンの行番号11の位置へ配置される。

比例法では既述のように，線幅を制御することはできないが，線分の比例配置法では，図3.16に□，■印で示される線分部の写像関数を定義することによって制御することが可能となる。

また，同図で，漢字パターンをアナログパターンとみた時の理想的な相似変換を示す破線からのずれが，変換漢字パターンでの文字バランスの劣化を表わすことになる。したがって，図からも明らかなように，線分の比例配置法では文字バランスの劣化は線幅と最小線分間隙線分部の変換において，その変換率に応じて一意的に生じるため，線分部の変換を出来るだけ比例法に近い結果となるように写像関数を作ることにより，字種対応に文字バランスの劣化を最小とすることができる。

(4) 変換処理

図3.17に線分の比例配置法による変換処理の流れを示す。

処理は次の4つの部分から成る。

① 線分の抽出

② 線分の比例配置

③ 線分間隙の処理

④ 行，列パターンの写像処理

(i) 〔ステップⅠ〕線分の抽出

(A) 原漢字パターンに対して，式(3.7)で定義される周辺分布X(j)，Y(i)と式(3.8)で定義される変化画素分布DX(j)，DY(i)を作成する。

(B) 周辺分布X(j)，Y(i)と変化画素分布DX(j)，DY(i)を用いて線分を含む行および列を

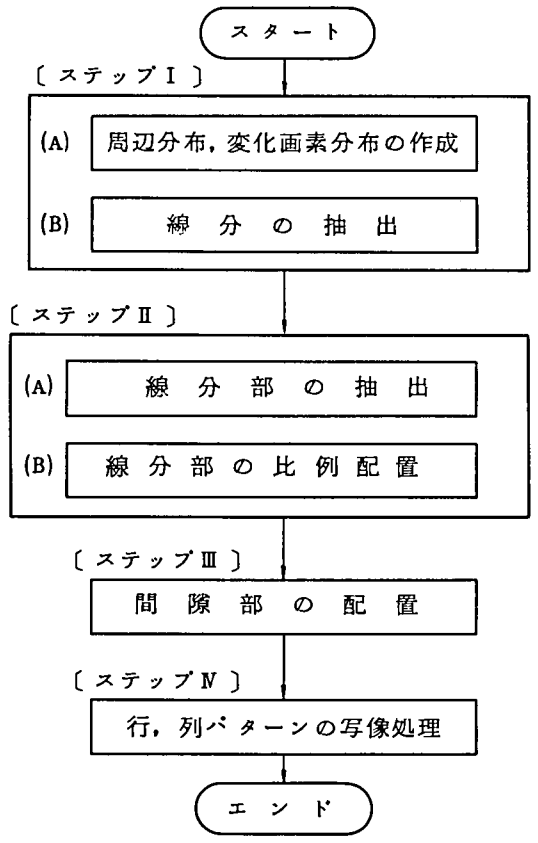


図3.17 線分の比例配置法の処理の流れ

抽出する。

(B)における線分を含む行，列の具体的な抽出処理は以下の通りである。（以下，縦線分の抽出処理を例に説明するが，横線分の抽出も同様にして行われる。）

① 候補列の抽出

周辺分布$X(j)$を幅w（原漢字パターンの線分幅）のバンドでjに関して走査を行ない，バンドの両端が共に隣接する列の周辺分布値より閾置T_Hより大きくなるとき，バンド内に含まれる列を縦直線を含む列の候補として選び出す。

すなわち，

$$X(j_0)>X(j_0-1)+T_H$$

かつ，

$$X(j_0+w-1)>X(j_0+w)+T_H$$

なら，列番号

$$j_0, j_0+1, j_0+2, \cdots, j_0+w-1$$

の列が候補列となる。

但し，

$$1\leq j_0 \leq n-w$$ （nは原漢字パターンの横サイズ）

$X(0)=0$，$X(n+1)=0$である。

図3.9の例では$T_H=3$，$w=1$に対して番号5，15，18の3列が選ばれる。

② 線分の確認

前述の処理で選出された候補列に対して変化画素分布$DX(j)$を用いて線分を確認する。

すなわち，変化画素分布$DX(j)$をjに関して走査し，候補列として選出された連続するw個の列の左端で分布値に閾値T_D以上の増加があれば，その連続する列は幅wの縦直線を含む列であると判断し，そうでなければ直線は含まないものとする。

図3.9の例では$T_D=3$に対して，番号5，15，18の列がいずれも縦線を含む列として抽出される。

(ii) 〔ステップⅡ〕線分の比例配置

(A) 線分部の抽出

ステップⅠで抽出された線分から線分間隙を調べ，原漢字パターンにおいて定められた最小線分間隙であれば，その間隙をはさむ線分を含めて線分部として抽出する。

(B) 線分部の比例配置

原漢字パターンと変換漢字パターンとの間での行および列番号の写像関数F(j)を線分部（線分と最小線分間隙を含む行および列番号部）と間隙部（最小線分間隙以上の幅をもつ線分間隙を含む行および列番号部）とに分けて，次のように定義する。

$$F(j)=\begin{cases} G_\alpha(j)\ ;\ \text{for}\ j_\alpha \leqq j \leqq j_\alpha+g_\alpha-1\ \text{に在る間隙} \\ L_\alpha(j)\ ;\ \text{for}\ j_\alpha \leqq j \leqq j_\alpha+l_\alpha-1\ \text{に在る線分部} \end{cases} \qquad (3.17)$$

ここで，g_α，l_α は線分間隙幅と線分部の幅である。

すなわち，$G_\alpha(j)$は列番号j_α，$j_\alpha+1$，$j_\alpha+g_\alpha-1$の列にわたって存在する間隙部（以後，第j_α 番の列から始まって幅g_α の間隙を間隙部αと略記する）に対する写像関数であり，$L_\alpha(j)$は同様に列番号j_α，$j_\alpha+1$，$j_\alpha+l_\alpha-1$の列にわたって存在する線分部（以後，間隙の表記にならって線分部αと記す。）に対する写像関数である。

a) 縮小変換の場合

線分部αの写像関数$L_\alpha(j)$は，さらに，次のように表わされる。

$$J=L_\alpha(j)=T_{l_\alpha}(j-j_\alpha+1)+\Delta(j_\alpha) \qquad (3.18)$$

$$j_\alpha \leqq j \leqq j_\alpha+l_\alpha-1$$

式（3.18）の第1項の関数$T_l(k)$は$1 \leqq k \leqq l$の範囲で定義され，原漢字パターンと変換漢字パターンに対して設定される縦直線幅w，Wと最小線分間隙d，Dおよび原漢字パターン上での線分部の幅lによって一意的に決定される関数である。

一方，$\Delta(j_\alpha)$は線分部の位置j_αに応じて変換処理過程できめられる$j_\alpha \leqq j \leqq j_\alpha+l_\alpha-1$の範囲で一定値をとる定数である。

したがって表3.1に示した変換パラメータが設定され，また，ステップⅠおよびステップⅡの(A)において線分部の位置j_αと幅l_αが求まった後には，定数

$\Delta(j_\alpha)$を決定しさえすれば線分部αに対する写像関数$L_\alpha(j)$は決定される。

そこで，図3.16と式(3.18)を対応させることによって，定数$\Delta(j_\alpha)$がもつ意味を考えてみる。

式(3.18)の第1項の関数$T_\ell(k)$は，図3.16で□と■印の相対的な並びが成す形状を決定することにより，変換パラメータに従った線分幅と最小線分間隙幅の変換を実現しており，定数$\Delta(j_\alpha)$は□と■印の位置を決めることにより文字バランスを制御していることが分かる。

以上のことから文字バランスの劣化を最小にする条件から定数$\Delta(j_\alpha)$が求まる。

文字バランスの劣化を最小とするためには各線分部を単位として，その中に含まれる線分の位置を相似変換時の位置に最も近くなるように設定してやれば良い。すなわち，図3.16で□と■印をその相対的な位置を保って相似変換を示す破線の最も近くに設定してやれば良い。

このことから，次に示す値Sを最小とする条件で定数$\Delta(j_\alpha)$が求まる。

$$S=\left|\sum_{j=j_\alpha}^{j_\alpha+\ell_\alpha-1}(P(j)-L_\alpha(j))\right| \qquad (3.19)$$

ここで，P(j)は相似変換を与える写像関数

$$P(j)=\frac{N-1}{n-1}(j-1)+1 \qquad (3.20)$$

$$(1\leqq j\leqq n)$$

であり，また

$$L_\alpha(j)=T_{\ell_\alpha}(j-j_\alpha+1)+\Delta(j_\alpha)$$

$$(j_\alpha\leqq j\leqq j_\alpha+\ell_\alpha-1)$$

である。

変換パラメータn，N，それに線分部の位置j_αと幅ℓ_αが既知であれば式(3.19)で与えられるS値を最小とする$\Delta(j_\alpha)$が求まる。

特に，関数P(j)は列番号jに関する1次関数であるため式(3.19)で定義されるSを最小とする$\Delta(j_\alpha)$は各線分部毎に式(3.19)をまともに計算することなく次式で求めることができる。

$$\Delta(j_\alpha) = P(j_\alpha + k_\alpha - 1) - T_{l_\alpha}(k_\alpha) \qquad (3.21)$$

ここで，値 k_α は，

$$C = \left| \left\{ \left(\sum_{k=1}^{l_\alpha} T_{l_\alpha}(k_\alpha) \right) / l_\alpha \right\} - T_{l_\alpha}(k_\alpha) \right| \qquad (3.22)$$

を最小とする条件から求まる。

したがって，あらかじめ，各変換パラメータ値と線分部の幅に応じて，関数 $T_{l_\alpha}(k)$ 対応に式（3.22）で与えられる値Cを最小とする k_α を求めておくことにより，式（3.21）から定数 $\Delta(j_\alpha)$ が簡単に算出される。

b）拡大変換の場合

拡大変換，厳密には変換パラメータがW＞w又はD＞dの場合の原漢字パターンと変換漢字パターンの行，列番号の対応関係の例を図3.18に示す。この例では $w=2$，W＝3，$d=2$，D＝4である。

この例から分かるように拡大変換の場合には式（3.18）の関数 $T_l(k)$ は多価関数とならなければならず，写像を定義できない。

そこで，拡大変換の場合は関数 $T_l(k)$ がとるべき値が複数個ある場合には，その中の最も小さな値を取る1価関数 $T'_l(k)$ を定義して，改めて，原漢字パターンから変換漢字パターンへの写像関数とする。

すなわち，

$$J = L_\alpha(j) = T'_{l_\alpha}(j - j_\alpha + 1) + \Delta(j_\alpha) \qquad (3.23)$$

$$j_\alpha \leqq j \leqq j_\alpha + l_\alpha - 1$$

である。

$j_\alpha \leqq j \leqq j_\alpha + l_\alpha - 2$ の範囲で

$$L_\alpha(j+1) - L_\alpha(j) \geqq 2 \qquad (3.24)$$

の場合には変換漢字パターンの行，列番号Jを $L_\alpha(j) < J < L_\alpha(j+1)$ の範囲で補間することが必要である。

そこで，改めて

$$T'_{l_\alpha}(j_\alpha) \leqq J \leqq T'_{l_\alpha}(j_\alpha + l_\alpha - 1) \qquad (3.25)$$

の範囲で補間関数を定義し，それを

$$T_{l_\alpha}(j - j_\alpha + 1)$$

とする。

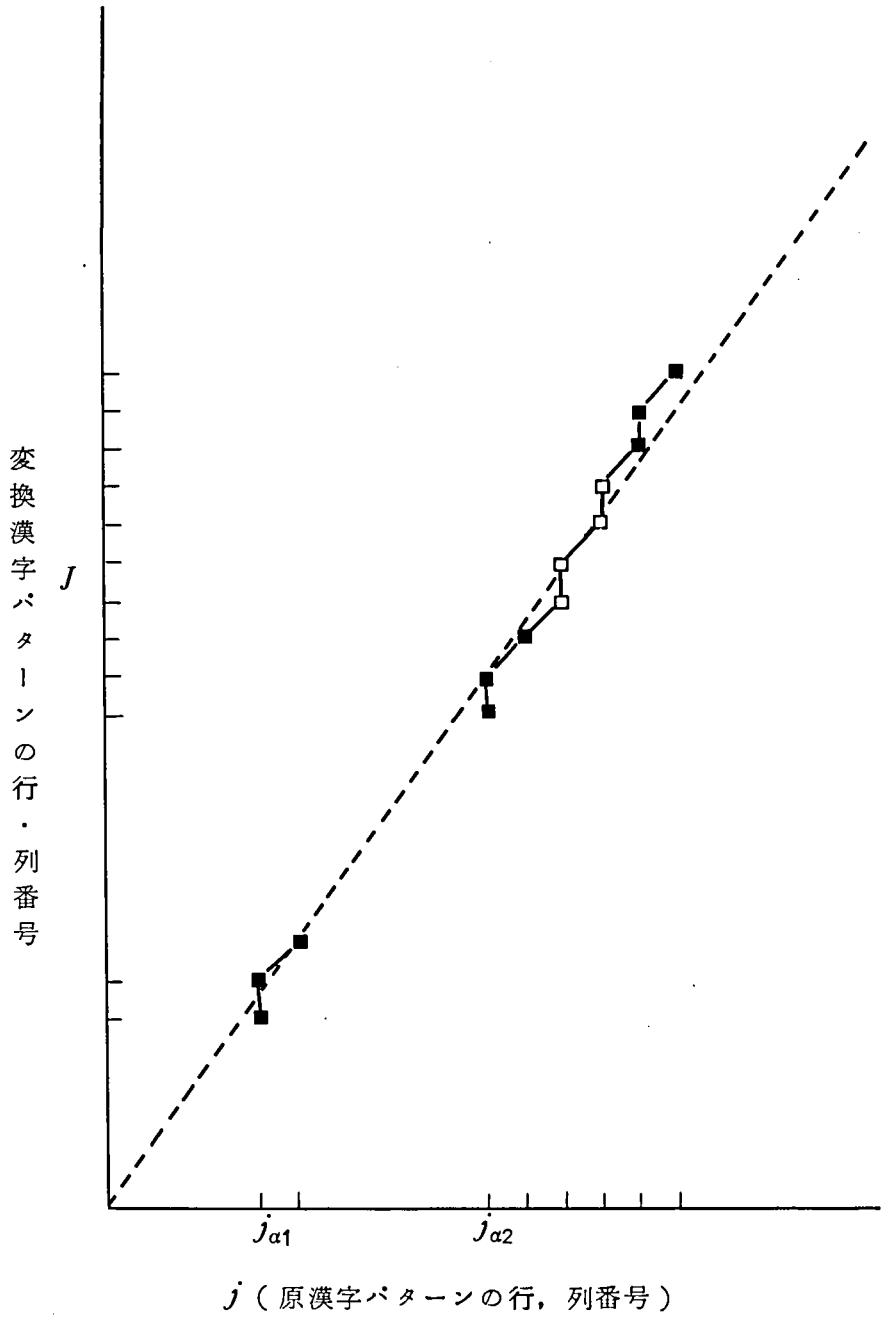


図 3.1 8　拡大変換の場合の線分部の行，列番号の対応関係

この時，$T_l(k)$ は $1 \leqq k \leqq l$ で定義された多価関数である。

関数 $T_l(k)$ は縮小変換の場合と同様に変換処理に先だって予め決定しておくことが出来る。

式（3.2 3）の $\Delta(j_\alpha)$ は次のように求められる。

$$\Delta(j_\alpha) = P(j_\alpha + k_\alpha - 1) - T_{l_\alpha}^{(n_\alpha)}(k_\alpha) \qquad (3.2\ 6)$$

ここで，$T_{l_\alpha}^{(n_\alpha)}(k_\alpha)$ は複数個の関数値 $\{T_{l_\alpha}(k_\alpha)\}$ の中の第 n_α 番目の値を意味する。

k_α，n_α は値

$$\left|\left(\sum_{k=1}^{l_\alpha}\sum_{n}T_{l_\alpha}{}^{(n)}(k)\right)\Big/\left\{\max_{n}\left(T_{l_\alpha}{}^{(n)}(l_\alpha)\right)-\min_{n}\left(T_{l_\alpha}{}^{(n)}(1)\right)\right\}-T_{l_\alpha}{}^{(n)}(k)\right| \quad \cdots\cdots (3.27)$$

を最小とする n，k として求まる。

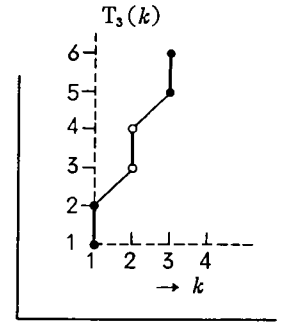
実際には変換処理に先立って補間関数 $T_l(k)$ を定めた時点で，k_α，n_α を求めておくことができる。

以上により線分部に対する写像関数が，変換パラメータに従った線幅と最小線分間隙幅をもち，文字バランスの劣化を最小にする条件を満足して決定されたことになる。

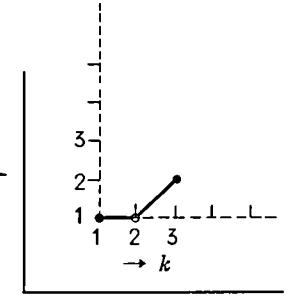
$T_l(k)$ の例を表 3.2 に示す。

表 3. 2　関数 $T_l(k)$ の例

変換パラメータ					$T_l(k)$								
変換前		変換後		l \ k	1	2	3	4	5	6	7	8	9
w	d	W	D										
1	1	1	1	1	1								
				3	1	2	3						
		2	2	1	1, 2								
				3	1, 2	3, 4	5, 6						
		3	3	1	1, 2, 3								
				3	1, 2, 3	4, 5, 6	7, 8, 9						
2	2	1	1	2	1	1							
				6	1	1	2	2	3	3			
		2	2	2	1	2							
				6	1	2	3	4	5	6			
		3	3	2	1	2, 3							
				6	1	2, 3	4	5, 6	7	8, 9			
3	3	1	1	3	1	1	1						
				9	1	1	1	2	2	2	3	3	3
		2	2	3	1	1	2						
				9	1	1	2	3	3	4	5	5	6
		3	3	3	1	2	3						
				9	1	2	3	4	5	6	7	8	9



$(w, d, W, D, l)=(1, 1, 2, 2, 3)$



$(w, d, W, D, l)=(3, 3, 2, 2, 3)$

(iii) 〔ステップⅢ〕間隙部の配置

ステップⅡの処理により，先ず，線分部に対する写像関数が優先的に決定された。そこで，次に，間隙部に対する写像関数を決定してやれば良い。

a) 縮小変換の場合

今，線分部βと線分部γに挟まれた間隙部αを対象にする。

この時，2点

$(j_\beta+\ell_\beta-1,\ L_\beta(j_\beta+\ell_\beta-1))$ と

$(j_\gamma,\ L_\gamma(j_\gamma))$

を直線的に結ぶ関数を求めれば良い。

すなわち，

$$G_\alpha(j)=\left[\frac{L_\gamma(j_\gamma)-L_\beta(j_\beta+\ell_\beta-1)}{j_\gamma-(j_\beta+\ell_\beta-1)}\cdot(j-(j_\beta+\ell_\beta-1))\right.$$

$$\left.+L_\beta(j_\beta+\ell_\beta-1)+0.5\right] \qquad \cdots\cdots(3.28)$$

$$(j_\alpha\leqq j\leqq j_\alpha+\ell_\alpha-1)$$

但し，〔　〕はガウス記号，ℓ_α，ℓ_β，ℓ_γは間隙部α，線分部β，γの幅である。

ここで，

$$j_\gamma=j_\alpha+\ell_\alpha$$

$$j_\beta+\ell_\beta-1=j_\alpha-1$$

であるから式(3.28)は次のようになる。

$$G_\alpha(j)=\left[\frac{L_\gamma(j_\alpha+\ell_\alpha)-j_\beta(j_\alpha-1)}{\ell_\alpha+1}\cdot(j-j_\alpha+1)+L_\beta(j_\alpha-1)+0.5\right] \qquad (3.29)$$

$$(j_\alpha\leqq j\leqq j_\alpha+\ell_\alpha-1)$$

以上により線分部αに対する写像関数が決定された。なお，漢字パターンの左，右両端が間隙部になる場合には

$$\left.\begin{aligned}&G_1(1)=1\\&G_*(n)=N\end{aligned}\right\} \qquad (3.30)$$

なる境界条件を用いることになる。

b）拡大変換の場合

線分部βと線分部γに挟まれた間隙部αを考える。

拡大変換の場合は，a）で述べた縮小変換の場合の座標系（j，J）を（J，j）座標系に変更し，2点

$$(T_{\ell\beta}^{(nmax)}(j\beta+\ell\beta-1)+\Delta(j\beta)+1,\ j\beta+\ell\beta)\ と$$
$$(T_{\ell\gamma}^{(1)}(1)+\Delta(j\gamma)-1,\ j\gamma-1)$$

を直線的に結ぶ関数を求めれば良く，縮小変換の場合と同様である。

(Ⅳ) 〔ステップⅣ〕行，列パターンの写像処理

ステップⅢまでで得られた写像関数を用いて原漢字パターンの行パターンと列パターンを変換漢字パターン上に写像する。

(5) シミュレーション

線分の比例配置法によるサイズ変換シミュレーションの結果を図3.19に示す。同図で(a)はサイズ（24×24）の原漢字パターン，(b)はサイズ（18×18）の縮小変換漢字パターン，(c)はサイズ（28×28）の拡大変換漢字パターンである。

変換パラメータは表3.3に示す通りである。

表3.3　シミュレーションで用いた変換パラメータ

パラメータ ＼ パターン		原漢字パターン	変換漢字パターン	
サイズ	横	24	18	28
	縦			
線分幅	横	1	1	
	縦			
最小線分間隙	横	1	1	
	縦			

シミュレーションに用いた原漢字パターンは写植用中細明朝体漢字パターンを（24×24）のメッシュに分割し，各メッシュ内の黒の含有率を閾値処理して2値パターンに変換した後，著者が修正して作成したパターンであり，デザイン上必ずしも完成したものではない。

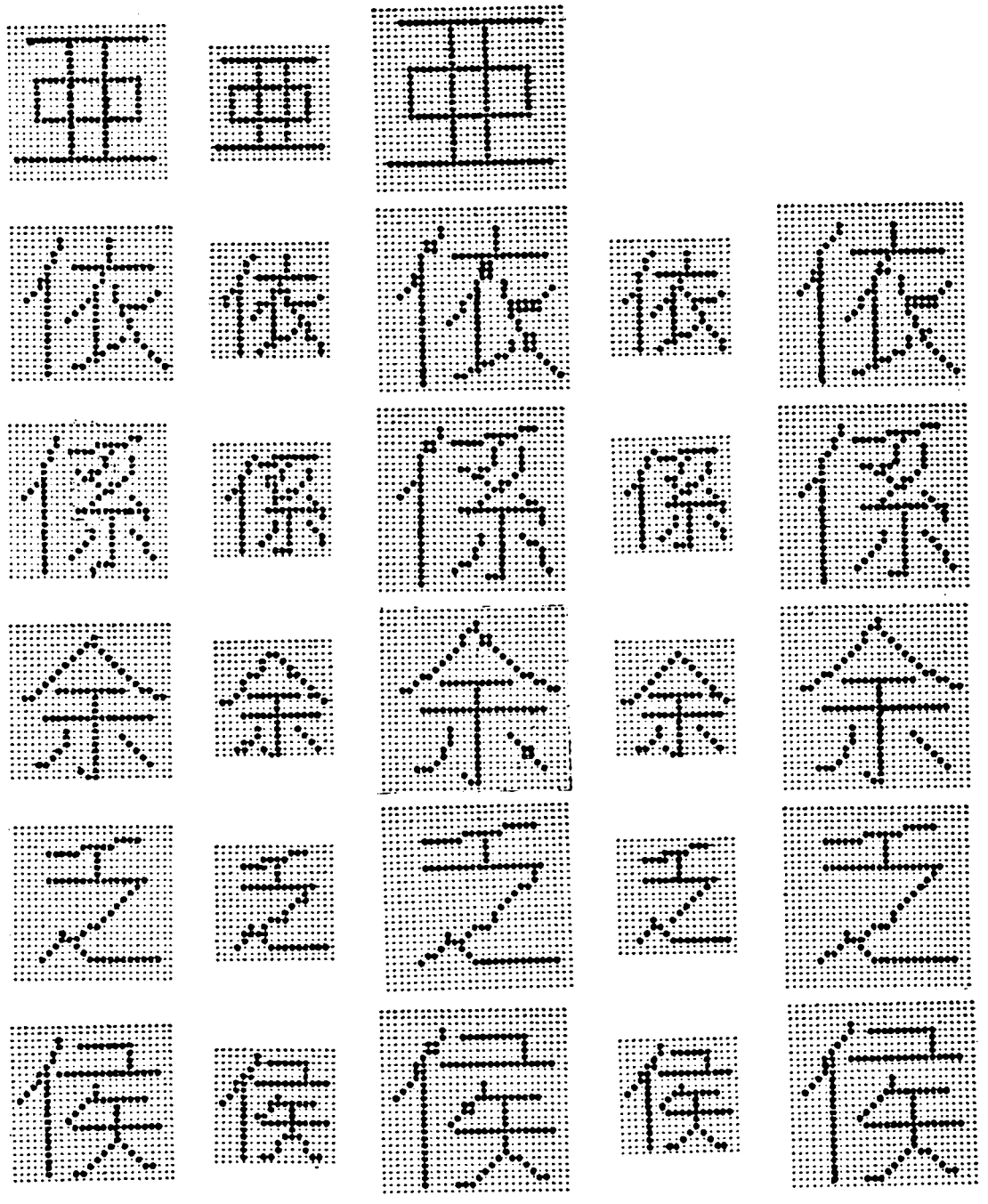


図 3.19の1　線分の比例配置法による変換漢字パターンの例

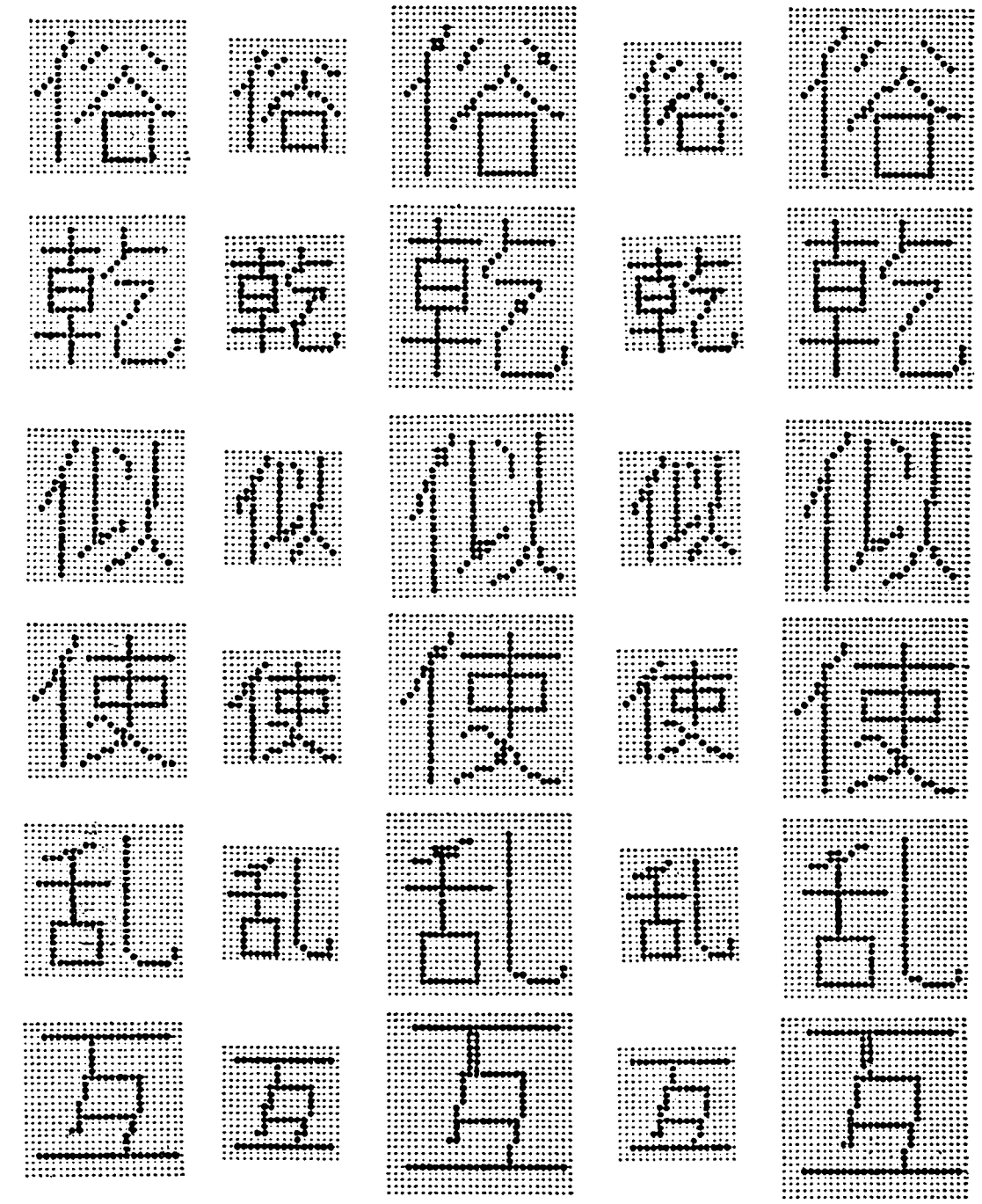


図 3.19の2　線分の比例配置法による変換漢字パターンの例

また，シミュレーション結果の出力は静電式のドットプリンタ上へ，黒画素は直径1㎜φ，白画素は直径0.2㎜φ，画素間隔1㎜で記録したものである。

また，線分の比例配置法では，縦，横直線のみから成る漢字パターンに対しては品質劣化が少ないことは明らかであるため，斜線，曲線を含む漢字パターンを中心にシミュレーション結果を示している。

まず，縦，横直線のみから成る漢字「亜」を見ると期待通りに直線幅も統一され，かつ，文字バランスの劣化もなく，満足すべきサイズ変換結果が得られている。

また，斜線，曲線を含む他の全ての変換漢字パターンにおいても，“行列選択法”や“比例法”の欠点であった，変換漢字パターンにおける文字バランスの劣化や，縦横直線幅の不統一は除去されており，線分の比例配置法に対する所期の改善効果を達成することができている。

しかし，「依」，「係」，「余」，「乏」などの長い斜線を含む変換漢字パターンにおいて，斜線，曲線部の画素配列の不自然さが目立ち，変換漢字パターンの文字品質を大きく劣化させていることが分かる。

そこで，次に線分の比例配置法での斜線の修正処理について考察する。

(6) 斜線の修正

3.2.5項で考察した行列選択法と線分の比例配置法とに共通して，行および列を単位とした画素の挿入，削除による漢字パターンのサイズ変換法は，縦および横方向の直線を多数含み，かつ，それらの直線が漢字パターンの構成上重要で，文字品質を支配しているとの立場に立った方法であった。したがって，斜線部に対する考慮は一切払われていなかった。この結果，図3.19に見られるように，文字識別上は決定的な防げにはならないまでも，斜線部の画素配列の不自然さが無視できないものとなる。

(i) 斜線の乱れの分類

画素による斜線の表示は斜線のこう配と線幅によって種々の表現形式がとられ一意的に定まっていない。しかし，斜線の表現形式が定まっていれば，画素配列の乱れは，その位置さえ検出できれば，修正は容易に行なわれる。したがって，画素の乱れの位置の検出が問題となる。

行列選択法あるいは線分の比例配置法では斜線部の画素の乱れは行および列の挿入，削除によって生じるものであるから，挿入，削除を行なった行および列の位置から画素の乱れの位置が検出できる。

ここでは表現形式が一意的で，それだけ画素の乱れが目立つ，勾配が±１の斜線を対象に挿入，削除を行なった行，列の交点（ i_c, j_c)から斜線までの距離を用いて，斜線の乱れを分類する。ここで，挿入，削除した行，列とは，変換漢字パターン上で，重畳あるいは反復挿入された行，列の組の中で最も若い番号をもつ行，列を意味する。また，点（ i_c, j_c)と斜線との距離とは点（i_c, j_c）から縦あるいは横方向に走査して斜線に達するまでに走査した画素の個数である。

① 縮小サイズ変換の場合

乱れＳ１；距離１で生じる乱れ

乱れＳ２；距離２で生じる乱れ

② 拡大サイズ変換の場合

乱れＫ０；距離０で生じる乱れ

乱れＫ１；距離１で生じる乱れ

図３.２０に上記の斜線の乱れの例を示す。

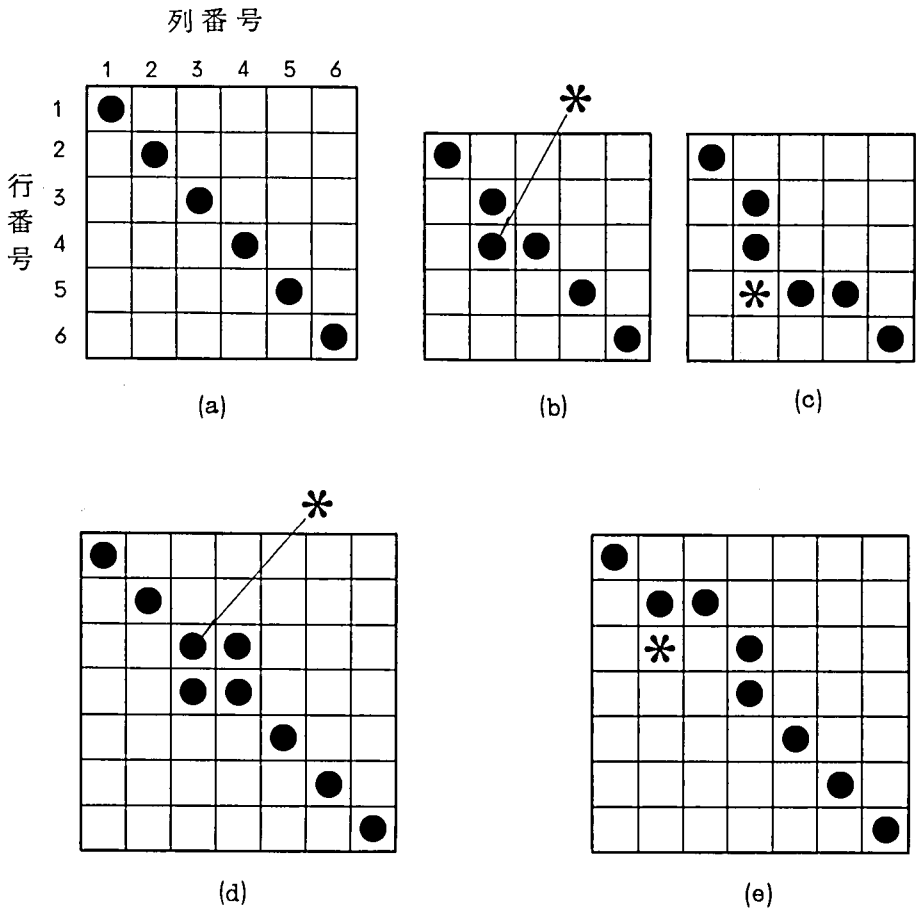


図 3.２０　斜線の乱れの例

同図で(a)は原パターンである。なお挿入，削除した行，列の交点（i_c，j_c）を＊印で示している。(b)は乱れＳ１の例で原パターンで第３行と第４行および第２列と第３列をそれぞれ重畳して第３行と第２列を削除した場合である。(c)は乱れＳ２の例で第４行と第５行，第２列と第３列をそれぞれ重畳して第４行と第２列を削除した場合である。

また，(d)は乱れＫ０の例で第３行と第３列を反復し，第４行と第４列を挿入した場合であり，(e)は乱れＫ１の例で第３行と第２列を反復し，第４行と第３列を挿入した場合である。

なお，斜線の方向と距離に向きを考えることにより，乱れＳ１，Ｓ２，Ｋ１には４種類乱れＫ０には２種類の場合が存在する。

上に述べた４種の乱れ以外に距離が大きな場合に生じる斜線の乱れもあるが，その場合の乱れは１箇所に集中しないので視覚的に問題は少ないとして，修正の対象からはずすこととした。

(ii) 修正処理

図3.21に前記の４種類の乱れＳ１，Ｓ２，Ｋ０，Ｋ１のみを対象とした場合の修正処理の手続きを示す。

斜線の乱れの発生箇所が挿入，削除を行なった行と列の交点近傍に限定されることから，各交点ごとにそれを中心とする４ないし８個の画素状態を調べることにより，乱れの種類を識別して，所定の斜線表現形式に従った画素配列に修正する。

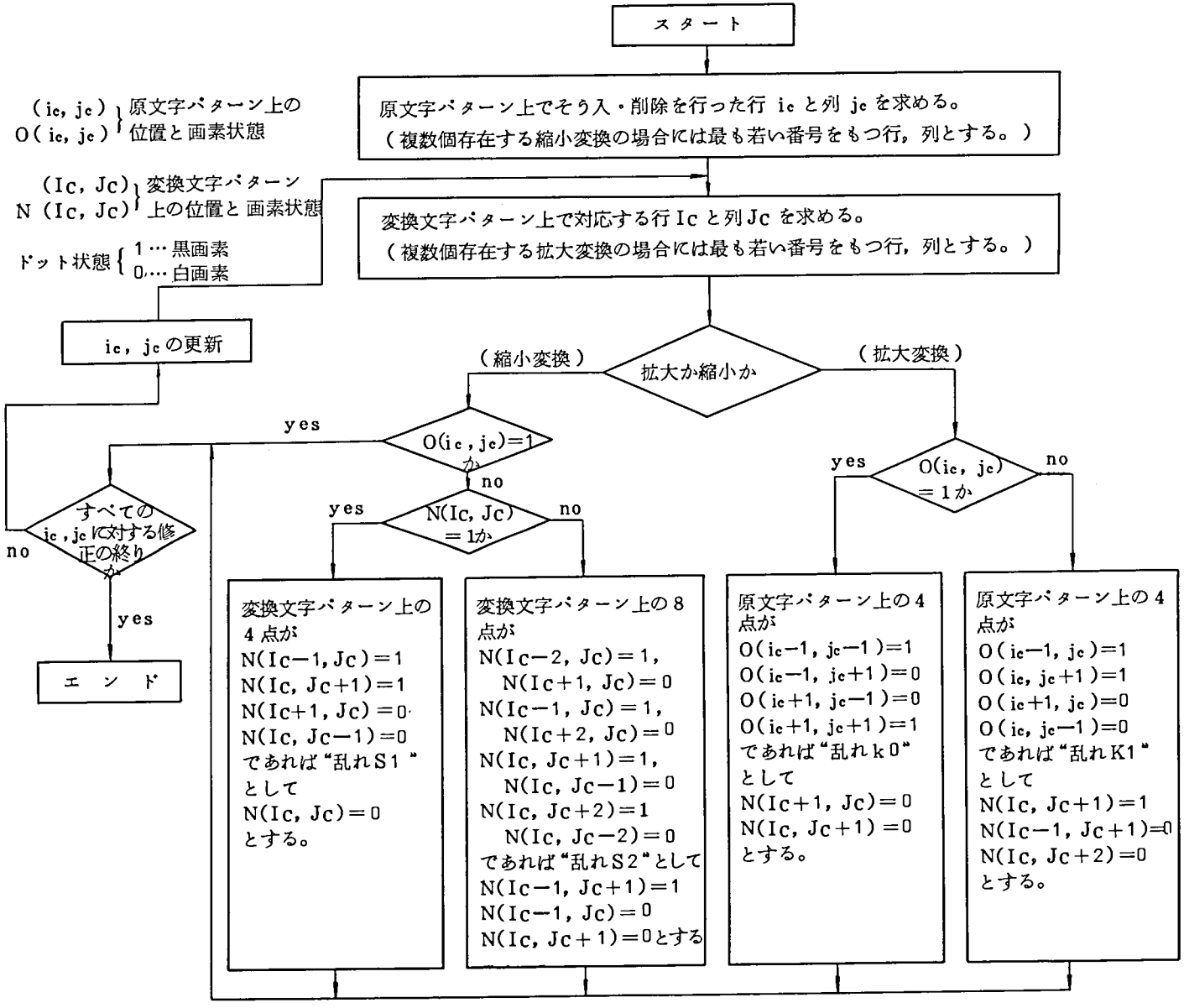


図 3. 2 1　斜線修正処理の流れ

(iii) 斜線の乱れの修正効果

斜線部の画素配列の乱れを修正した変換漢字パターンの例を修正前の変換漢字パターンと対比して図3.19に示す。同図で(d)は縮小変換漢字パターン，(e)は拡大変換漢字パターンである。

① 縮小変換漢字パターンに対する効果

乱れS1は漢字「依」，「余」，「乏」など斜線を含む変換漢字パターンに頻繁に現れ，乱れS2は漢字「余」，「侯」においてみられる。

縮小変換の場合には上記の2種類以外の斜線の乱れは無視することができ，かつ，これら2種類の斜線の乱れは，修正処理で完全に除去することができる。

② 拡大変換漢字パターンに対する効果

乱れK0は人偏「イ」の斜線部や「俗」，「余」，「乾」などに，又乱れK1は「似」，「使」，「係」，「乏」にみられる。これら2種類の乱れは，いずれも修正処理によって除去される。したがって，拡大変換漢字パターンに対しても，(i)に定義した2種類の乱れに関しては修正処理は十分有効であることが確認できる。

しかし，拡大サイズ変換の場合は乱れK0，K1以外の乱れが生じる。例えば，「乱」や「互」にみられるように，こう配が0に近い緩やかな傾きの斜線や，こう配が∞に近い急な斜線において斜線幅の表現が不自然となる。

図3.22に緩やかなこう配の斜線を例に周辺分布Y(i)と変化画素分布DY(i)を示す。斜線を含む3個の行の周辺分布値Y(4)，Y(5)，Y(6)は差が1ずつしかなく，(4)の(i)で述べた線分抽出処理では，この3行はいずれも横直線を含む行として抽出されず反復挿入が行なわれることになる。

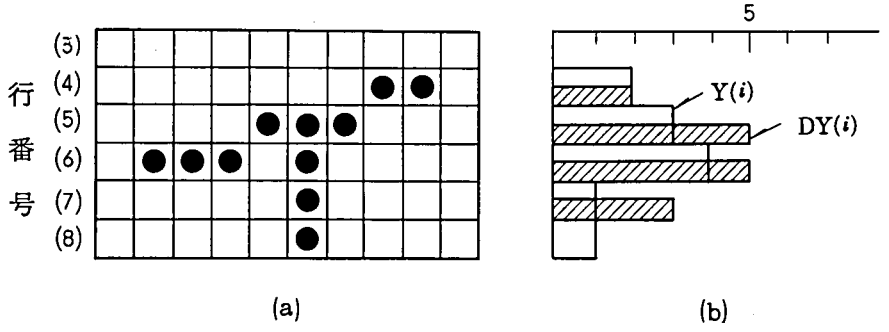


図3.22 緩やかな勾配の斜線に対する周辺分布Y(i)と変化画素分布DY(i)

このように緩やかなこう配の斜線や急なこう配の斜線では，それらの斜線を含む連続する行または列は，一般に周辺分布，変化画素分布，さらには最大連続黒画素分布のいずれにおいても類似した値をとり，線分の抽出が難しい。したがって，これらの斜線表現の不自然さを完全に修正することは困難であり，今後の残された問題である。

また，漢字「依」にみられる2画素幅の横直線も原漢字パターンでは直線として抽出できなかったために生じたものである。このような直線は拡大変換の場合に，原漢字パターン上では線分の長さが短かく検出できなかったものが，拡大サイズ変換の結果，長さが長くなり不自然さを増大することになる。

(7) 線分の比例配置法の処理量

シミュレーション実験から得られた，線分の比例配置法によるサイズ変換の処理量をダイナミックステップ数として表3.4に示す。

シミュレーションプログラムで使用した言語は，(4)で述べた線分および線分間隙の比例配置処理と斜線部の修正処理がフォートランであり，他は小形計算機ＰＤＰ－１１のアセンブラ言語である。

この結果，ステップＩの線分の抽出処理に約２９Ｋステップが必要となる。それ以降は，拡大，縮小のサイズ変換の方向によって異なる。縮小サイズ変換では，線分部の抽出と写像関数の作成に約８Ｋステップ（但し，フォートランの１ステップをアセンブラの５ステップに換算），変換漢字パターンの作成に約１０Ｋステップ，斜線部の修正処理に約２Ｋステップが必要となる。この結果，縮小サイズ変換処理では合計約５０Ｋステップの処理量となる。

一方，拡大サイズ変換処理では，写像関数の作成に約７Ｋステップ，変換漢字パターンの作成に約１７Ｋステップ，斜線部の修正処理に約２Ｋステップなる。この結果，拡大サイズ変換処理には合計約５５キロステップがかかることになる。

したがって，サイズ変換処理を平均１μ秒の命令実行時間のミニコンピュータで実行するとすれば変換速度は縮小サイズ変換の場合２０字／秒，拡大サイズ変換の場合約１８字／秒が得られることになる。

サイズ変換の処理量は原漢字パターンのサイズと変換倍率によって変化するが，周辺分布と変化画素分布の作成が主要な処理内容となる線分の抽出処理と変換漢字パターンの作成処理は，それぞれ原漢字パターンと変換漢字パターンのサイズの２乗に従

表3.4　線分の比例配置法のシミュレーションでのダイナミックステップ数

ステップ	処理内容	言語	ダイナミックステップ数 (Kstep)	備考
〔I〕	横周辺分布の作成	アセンブラ（PDP-11）	6.5	原漢字パターンのサイズは（24×24），16画素／語のパック形式での表現
	縦周辺分布の作成		6.0	
	横変化画素分布の作成		7.2	
	縦変化画素分布の作成		6.8	
	周辺分布の極大点検出		1.7	縦，横周辺分布の処理を合わせて
	変化画素分布による線分の確認		0.6	縦，横変化画素分布の処理を合わせて
〔II〕	線分部の抽出		0.7	縮小サイズ変換処理の場合のみ
〔III〕	縮小変換に対する写像関数の作成	フォートランIV	1.5	乗除算処理が多いためフォートランで作成
	拡大変換に対する写像関数の作成		1.4	
〔IV〕	縮小変換漢字パターンの作成	アセンブラ	10.4	縮小変換漢字パターンのサイズ（20×20）
	拡大変換漢字パターンの作成		17.3	拡大変換漢字パターンのサイズ（28×28）
	縮小変換の斜線部修正処理	フォートラン	0.4	サイズ（24×24）から（20×20）への変換
	拡大変換の斜線部修正処理		0.3	サイズ（24×24）から（28×28）への変換

って変化し，写像関数の作成処理はサイズに従って変化する。

サイズの変換方向による差異としては，線分部の抽出処理が，拡大サイズ変換処理において線分間隙の消失のおそれが無いため不要となる。その結果，写像関数の作成処理において線分部の作成処理が多小簡単となる。

また，斜線部の修正処理において，拡大サイズ変換の場合は原漢字パターン上で挿入した行，列の交点の画素と4連結関係にある4個の画素の状態から斜線の方向のみを識別すれば画素配列の乱れを特定することが出来るのに対して，縮小変換の場合は黒画素の密度を高める方向の変換であるため，削除した行，列の交点の画素を中心とした所定の距離の範囲内に非斜線部の構成黒画素の移入が生じるため，原漢字パターン上で黒画素配列の乱れを特定することはできず，変換漢字パターン上での識別処理が必要となる。

特に乱れS2の識別には8個の画素状態を用いる必要があり，このため，拡大変換に比べると縮小変換の場合が斜線部の修正処理は複雑となる。

(8) サイズ変換の範囲

図3.23に斜線修正処理を含めて，サイズ（16×16）から（32×32）までの範囲でサイズ変換を行なった変換漢字パターンの例を示す。

このサイズ変換範囲は，4本／㎜の分解能をもつ表示装置上では4㎜角から8㎜角の表示文字サイズの範囲となり，12ポイントから24ポイントの範囲に相当する。一般読み物，論文の印刷では本文用から大見出し用活字の大きさをカバーすることになる。

サイズ変換の限界としては，縮小サイズ変換の場合は漢字パターンの表現に最低限必要とされる（16×16）程度と考えられるが，拡大サイズ変換に対しては，3.2.4項で述べた比例法による整数倍の拡大サイズ変換と組み合わせることにより極めて広汎なサイズ変換が可能となる。なお，整数倍の変換に限定すれば比例法における欠点である直線幅の不揃いは生じない。

整数倍の変換処理と組み合わせて任意のサイズ変換の変換漢字パターンを得るために，非整数倍のサイズ変換処理がカバーすべき変換倍率を求めると次のようになる。

原漢字パターンのサイズをm_0，所要の変換漢字パターンのサイズをmとする。さらに，総合変換倍率Rを整数倍変換の変換倍率αと非整数倍変換の変換倍率βの積として，

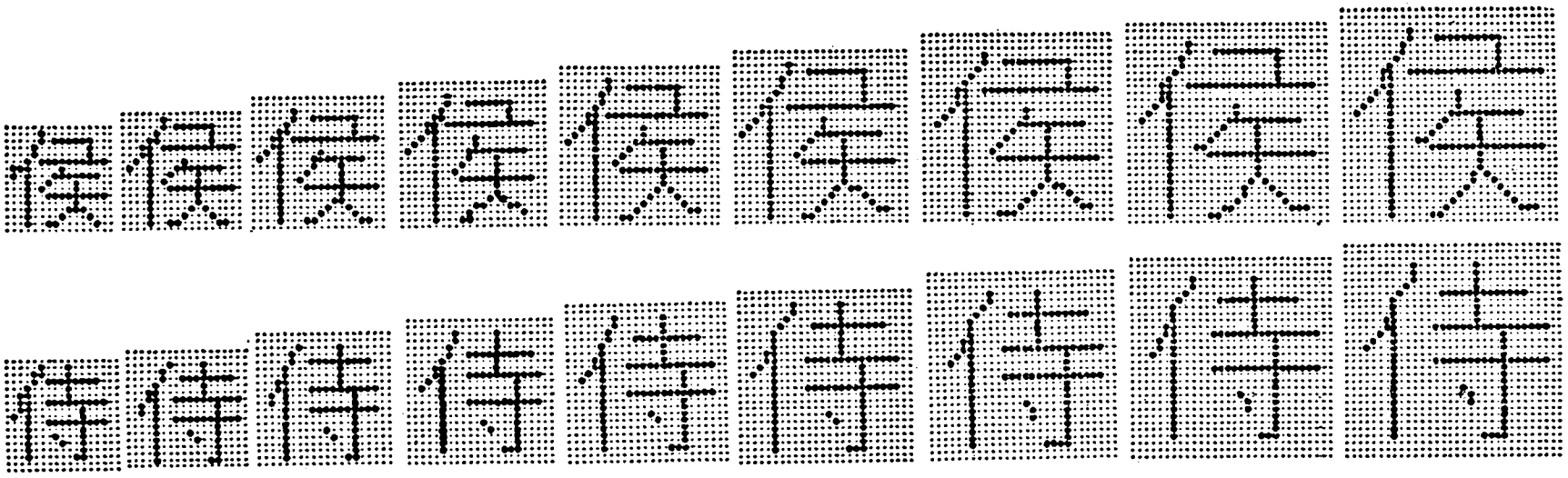


図 3.2 3　線分の比例配置法による拡大，縮小サイズ変換例

$$R = \alpha \cdot \beta \qquad (3.3.1)$$

と表わす。

(i) 拡大サイズ変換の場合

$$\alpha \cdot m_0 < m < (\alpha + 1)\, m_0 \qquad (3.3.2)$$

α：整数

が成立する整数倍変換が可能である。したがって，その後，非整数倍の拡大サイズ変換処理で所要のサイズ m を得る場合には，式（3.3.2）を

$$\alpha \cdot m_0 < m < \left(1 + \frac{1}{\alpha}\right) \cdot (\alpha \cdot m_0) \qquad (3.3.3)$$

と変形することにより変換倍率 β の範囲として

$$1 < \beta < \left(1 + \frac{1}{\alpha}\right) \qquad (3.3.4)$$

α：整数

となる。

(ii) 縮小サイズ変換の場合

$$\frac{1}{\alpha+1} \cdot m_0 < m < \frac{1}{\alpha} \cdot m_0 \qquad (3.3.5)$$

を満たす整数分の1の縮小変換が可能である。

式（3.3.5）を

$$\frac{\alpha}{\alpha+1} \cdot \left(\frac{m_0}{\alpha}\right) < m < \frac{m_0}{\alpha}$$

と変形することにより

$$\frac{\alpha}{\alpha+1} < \beta < 1 \qquad (3.3.6)$$

が得られる。

したがって，非整数倍のサイズ変換処理によって，変換倍率 β を

$$\frac{1}{2} < \beta < 2 \qquad (3.3.7)$$

の範囲で実現できれば，拡大，縮小いずれのサイズ変換処理によっても任意のサイズの変換漢字パターンを得ることが出来ることになる。

特に，変換倍率の大きな拡大変換を行なう場合，非整数倍変換の変換倍率βを出来るだけ小さくして，変換漢字パターンの文字バランスの劣化を少なく抑えようとすれば，所要の非整数倍の変換倍率βは次のようになる。

(iii) $\frac{1}{\gamma}<\beta<\gamma$で全範囲をカバーする場合

すなわち，$1/\gamma$に縮小する場合とγ倍に拡大する場合が同じ文字品質の劣化を生じる場合である。

$$\alpha \cdot m_0 < m < (\alpha+1)\, m_0, \quad \alpha：整数$$

の変換次数mを得る場合

$$(\alpha \cdot m_0) \cdot \gamma = (\alpha+1) \cdot m_0 \cdot \frac{1}{\gamma}$$

とおいて

$$\gamma = \sqrt{\frac{\alpha+1}{\alpha}} \qquad (3.3.8)$$

が得られる。αを整数とするとγは$\alpha=1$で最大値をとり，この時，非整数倍の変換倍率βは，

$$\frac{1}{\sqrt{2}} \leqq \beta \leqq \sqrt{2} \qquad (3.3.9)$$

となる。

(iv) 挿入，削除する行，列の個数を等しくする場合（$\beta_1 \leqq \beta \leqq \beta_2$）

すなわち，挿入，削除する行，列の個数が等しい時，文字品質の劣化が等しい場合である。

この時，

$$(\alpha+1) \cdot m_0 \cdot \beta_1 = (\alpha + \frac{1}{2}) \cdot m_0$$

$$\alpha \cdot m_0 - \beta_2 = (\alpha + \frac{1}{2}) \cdot m_0$$

であるから，

$$\left.\begin{aligned} \beta_1 &= (\alpha+\tfrac{1}{2})/(\alpha+1) \\ \beta_2 &= 1 + \frac{1}{2\alpha} \end{aligned}\right\} \qquad (3.4.0)$$

となる。

$\alpha=1$でβ_1は最小値，β_2は最大値をとり，結局，所要の変換倍率は

$$0.75 < \beta < 1.5 \qquad (3.41)$$

となる。

これらの変換倍率は線分の比例配置法によって充分実現可能な範囲である。

比例法による大きな倍率での整数倍変換を実行すれば，結果的には画素を拡大した変換パターンが得られることになり，斜線部に対する平滑化処理が必要となる。整数倍拡大変換に対する平滑化処理は黒崎[4-3]，大川[34]らによって研究されている。

なお，縦あるいは横の1方のみのサイズ変換処理を行なうことにより，字体を変換し平体，長体の漢字パターンを作成することも可能である。

図3.24は，インタレース表示方式において生じるフリッカーを無くすように，縦直線の線幅を1画素，横直線の幅を2画素となるように長字体に変換した漢字パターンの例であり，現在，電電公社で試験中の画像応答システム（ＶＲＳ）用の漢字パターン発生器内に組み込まれている。

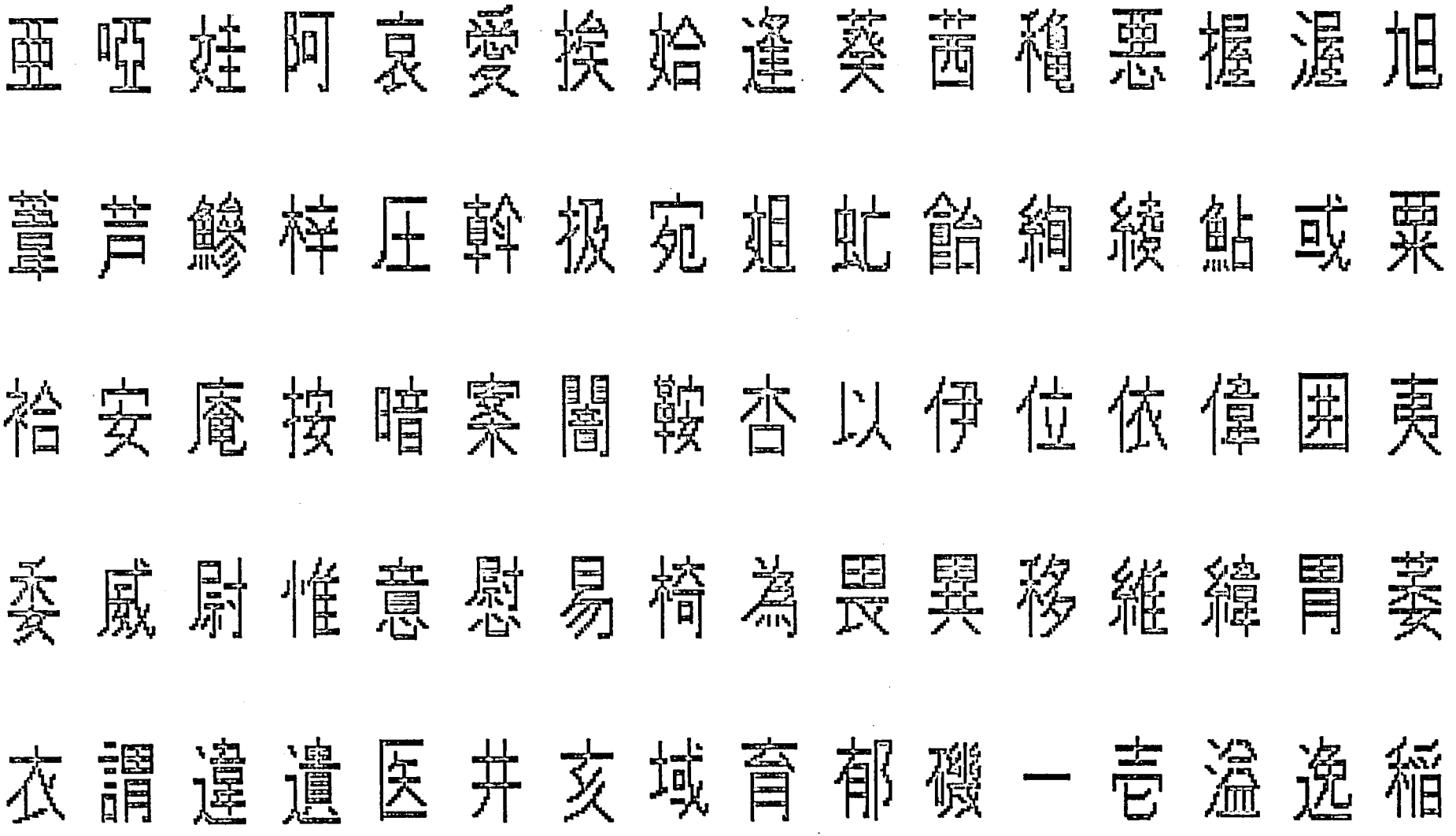


図 3.2.4　サイズ変換処理を用いて作成した長体の漢字パターン例(1)

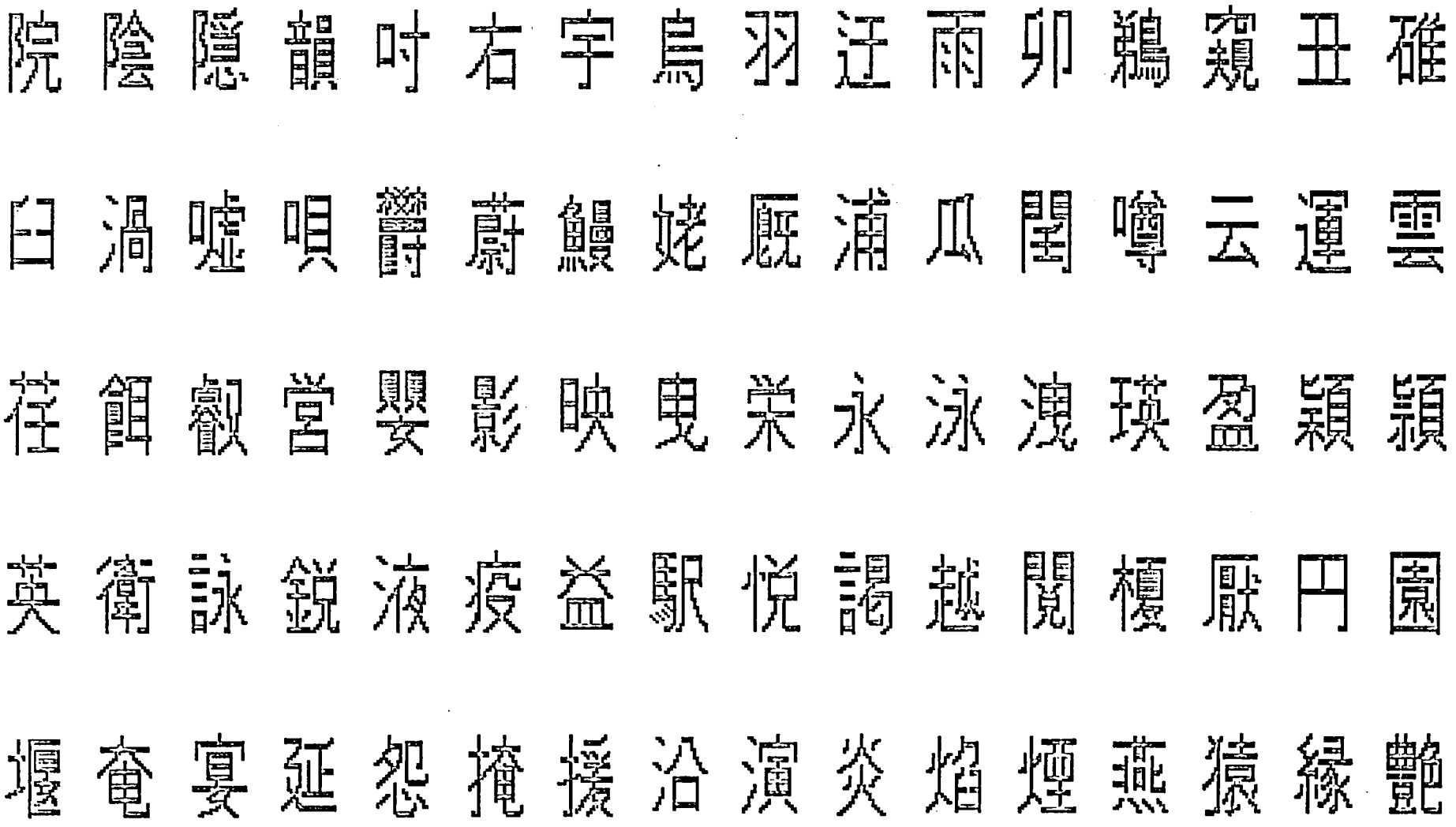


図 3.2 4　サイズ変換処理を用いて作成した長体の漢字パターンの例(2)

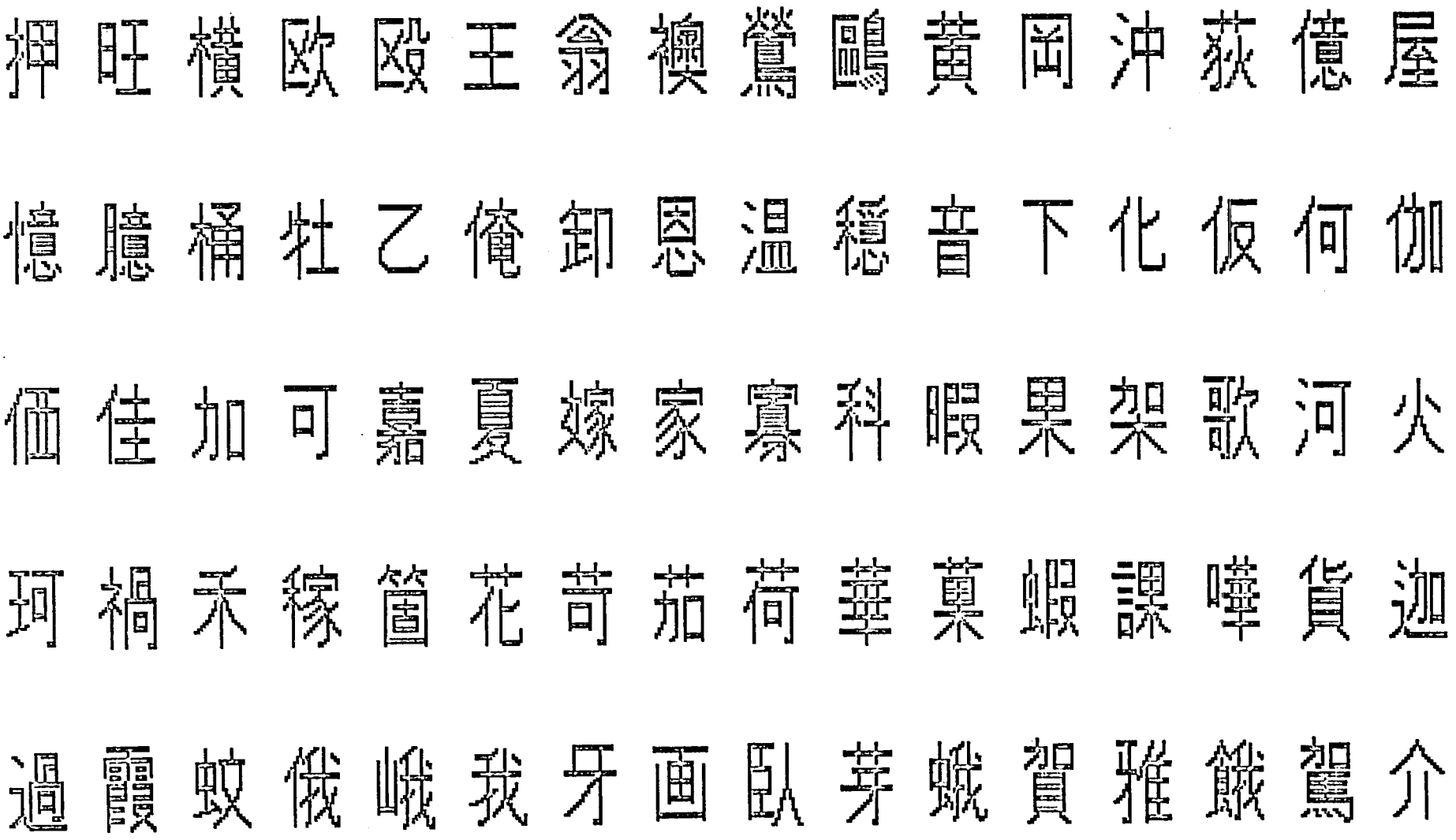


図 3.2 4　サイズ変換処理を用いて作成した長体の漢字パターン例(3)

3.4 サイズ変換処理の高速化

前節で述べた線分の比例配置法は字種対応に挿入，削除すべき行および列を選択することにより，直線幅を統一し，文字バランスの劣化を抑えて，かつ，非整数倍の拡大および縮小サイズ変換を可能とした。

線分の比例配置法では，挿入，削除すべき行および列の選択が自動的に可能であるため，サイズ変換のために付加すべき新たな制御情報は一切必要としない。したがって，図表を含んだ文書の編集時など，多くの種類の文字サイズが要求され，かつ，その文字サイズを予め予測できない場合など非常に有力なサイズ変換技術となる。

しかし，一方，各文字対応に3.3.2項で述べた写像関数を求める処理が必要であり，変換速度が遅いという問題がある。

そこで，高い頻度で用いられる変換漢字パターンのサイズを予じめ複数種類設定し，それらのサイズ変換を実現するために挿入，削除すべき行および列の識別情報をサイズ変換のための制御情報として原漢字パターンに付加しておく方法が著者ら[65]と渡部ら[30]と時を同じくして提案された。

この制御情報付与方式は制御情報に従って行および列を反復挿入あるいは重畳削除するだけで良く，変換処理が単純化され，装置化も容易となる。実際，高速漢字プリンタに実装されている例が見られる。[31]

この方式では，変換漢字パターンの種類が限定され，かつ，変換漢字パターン毎に新たな制御情報を必要とするため，漢字パターンのデータ圧縮方式の1つと見なすことも出来る。

以下，本節では制御情報の持ち方，情報量，制御情報の作成法について考察する。

3.4.1 制御情報と情報量

制御情報の与え方としては，前節で述べた全自動変換処理のどの段階まで前処理として実行しておくかで次のような場合が考えられる。

① 縦，横直線を含む行および列とそれ以外の行および列の識別情報（直線識別情報付与方式）

② 挿入，削除すべき行および列とそれ以外の行および列の識別情報（行列選択情報付与方式）

③ 行列選択情報に加えて，斜線部の乱れなどの修正用の画素位置情報（行列および画素選択情報付与方式）

(i) 直線識別情報付与方式

この場合には，原漢字パターン上での縦，横直線の抽出処理だけをあらかじめ実行しておくものであり，各変換倍率に応じて挿入，削除する行および列の選択は変換要求が生じた時点で行なう必要がある。

この場合には，制御情報は変換倍率に依らず，原漢字パターン毎に１種類ずつ付与すれば良い。

すなわち，各行および列について，それが縦，横の直線を含むか否かを１ビットを用いて符号化することにより，サイズ（$m \times n$）の原漢字パターン１個について（$m+n$）ビットの制御情報があれば良いことになる。

しかし，サイズ変換実行時に原漢字パターンと変換漢字パターン間の行，列番号に関する写像関数を作成することになり，サイズ変換処理の高速化が不充分となる。

そこで，本研究では行列および画素選択情報付与方式について検討する。

(ii) 行列および画素選択情報付与方式

この方式では変換漢字パターン毎に挿入，削除の対象となる行，列の選択情報と画素単位での修正情報が必要となるが，サイズ変換時には行，列の選択情報にしたがって行および列の反復挿入あるいは重畳削除を行なった後，修正位置の画素状態を反転させるだけで良く，高速なサイズ変換が可能となる。さらに，画素単位での修正をも可能であるため，高品質の変換漢字パターンが得られる。

以下，ここでは記録用として作成したサイズ（３２×３２）の漢字パターンを原漢字パターンとして，表示用のサイズ（１８×１６）の漢字パターンをサイズ変換処理により発生させる場合を具体的に設定し考察する。

変換漢字パターンのサイズを（１８×１６）と特定した場合は以下のような選択情報の付与方法が考えられる。

〔方法Ａ〕 削除する行と列の識別情報を全て制御情報として付与する。

〔方法Ｂ〕 横サイズ変換についてみると，原漢字パターンの横サイズが３２，変換漢字パターンのサイズが１６と丁度２対１であり，かつ，縦直線の線幅も２対１となる。

そこで，列については無条件に１列おきに削除し，行に関してのみ選択

情報を付与する。

〔方法C〕　列は無条件に1本おきに削除し，行は1本おきに選ばれる16本のうち2本を残し，14本を削除することとし，残す2本の行の識別情報のみを制御情報として付与する。

なお，上記の各方法について，行と列を削除し縮小サイズ変換を行なった後，画素単位の修正を行なう。

制御情報を

① 方法A，Bでの行，列の識別には各32ビットを用い，削除する行，列を1で，残す行，列を0で示す。

② 画素単位の修正位置は変換漢字パターン上の2次元座標で指定し，各座標値は2進符号で与える。

③ 方法Cの行の識別は16本中の順位を2進符号で示す。

の方法で構成することにすると，前記各法の制御情報量は次で与えられる。但し，画素単位での修正箇所の個数をNとする。

〔方法A〕

制御情報量 $Q_A = (4+5) \cdot N_A + 64$　　(3.42)

$= 9 \cdot N_A + 64$

〔方法B〕

制御情報量 $Q_B = 9 \cdot N_B + 32$　　(3.43)

〔方法C〕

制御情報量 $Q_C = 9 \cdot N_C + 8$　　(3.44)

画素単位での修正箇所の個数N_A，N_B，N_Cは削除する行，列の選択の自由度に応じて異なり，自由度の減少に伴ない，方法A，B，Cの順で修正箇所は増加する。

表3.5に各方式における修正点数の分布を示す。また，

表3.5　各方法における修正点数の分布

方法＼修正点数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	99	32	25	13	8	5	2	2	1	0	1	0	0	0	0
B	48	27	27	21	21	15	8	9	4	2	3	1	1	0	1
C	40	26	25	18	26	11	14	12	3	7	5	0	0	1	0

表3.6に１８８文字を対象に行なった画素単位での修正箇所の個数と制御情報量の比較実験結果を示す。

表3.6　各方法の制御情報量の比較

項目 / 方法	最大修正点数	平均修正点数	固定長方式の制御情報量(ビット／文字)	可変長方式の制御情報量(ビット／文字)
A	１０	1.18	１５４(0.53)*	74.62(0.26)*
B	１４	2.82	１５８(0.55)*	57.38(0.19)*
C	１３	3.22	１２５(0.43)*	36.98(0.13)*

＊　括弧内は(18×16)の漢字パターンのデータ量に対する制御情報量の割合

同表で固定長方式とは各方法で出現した最大の個数の修正箇所の指定に要する制御情報量であり，可変長方式とは平均個数の修正箇所の指定に要する制御情報量を意味する。

なお，可変長方式の制御情報量には，区切りを示す符号量は含めていない。

また，方法Ａ，Ｂ，Ｃによる縮小サイズ変換漢字パターンの例を図3.25に示す。

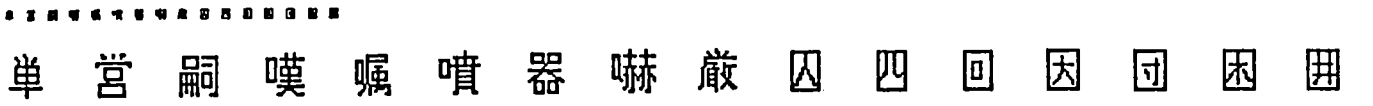
実験は修正箇所の個数には制限を付けずに行なったものであるが，最大１０～１４であった。しかし，多くの文字に対して，画素単位での修正の必要はなく平均修正箇所は少ないことがわかる。

この結果，可変長方式での制御情報量は，サイズ(１８×１６)の漢字パターンをそのまま表現する場合の情報量に比較して，方法Ａ，Ｂ，Ｃでそれぞれ0.26，0.19，0.13倍と少なくて済むことがわかる。

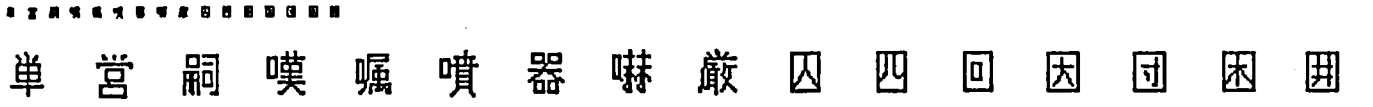
変換漢字パターンの品質については方法Ａは充分実用的な品質が得られる。方法Ｂ，Ｃでも大多数の文字に対しては充分実用に耐えられるが，列の選択の自由度がないため１部の文字で線分間隙の消滅による誤字が発生する。

原漢字パターン(32×32)

〔方法A〕



〔方法B〕



〔方法C〕

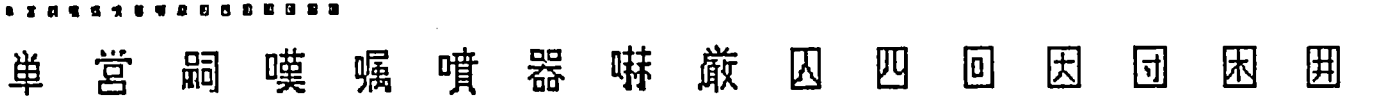


図 3.2 5　行列および画素選択情報付与方式による変換パターン例(1)

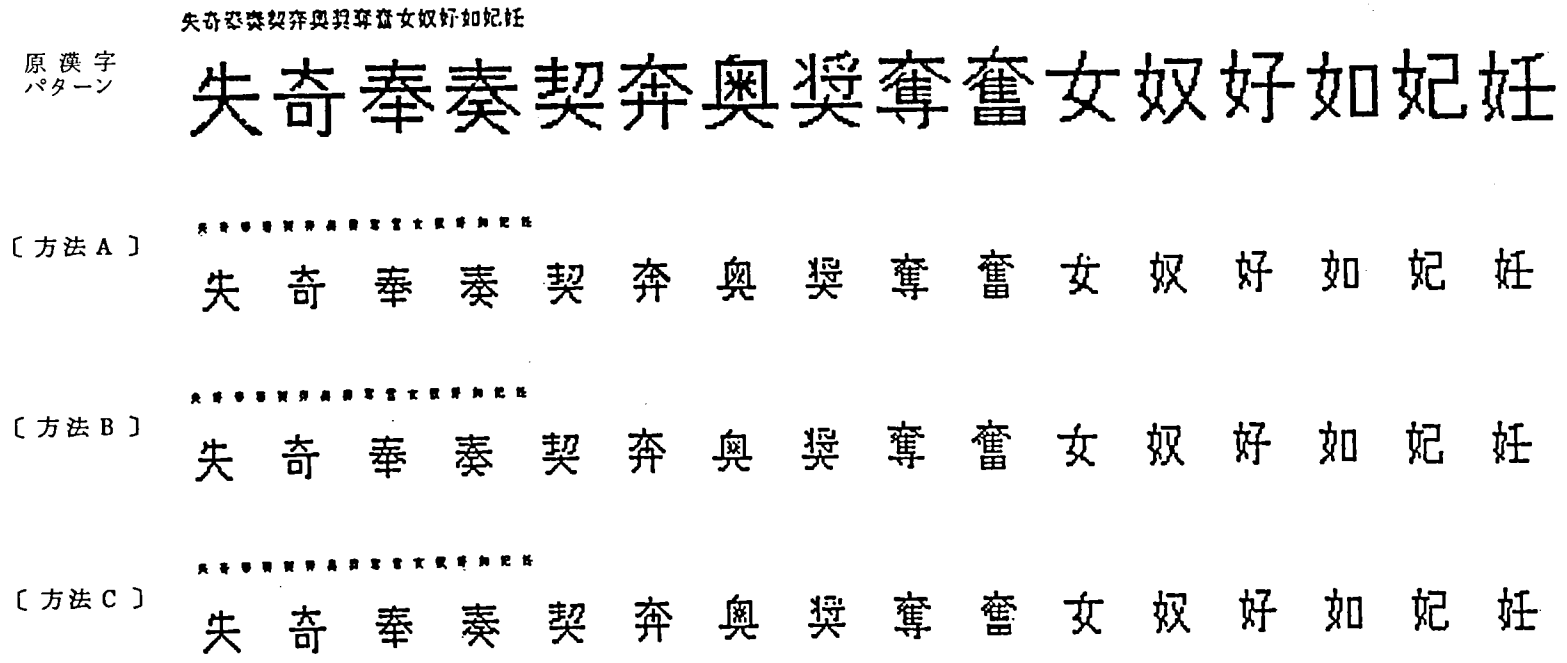


図3.25　行列および画素選択情報付与方式による変換パターン例(2)

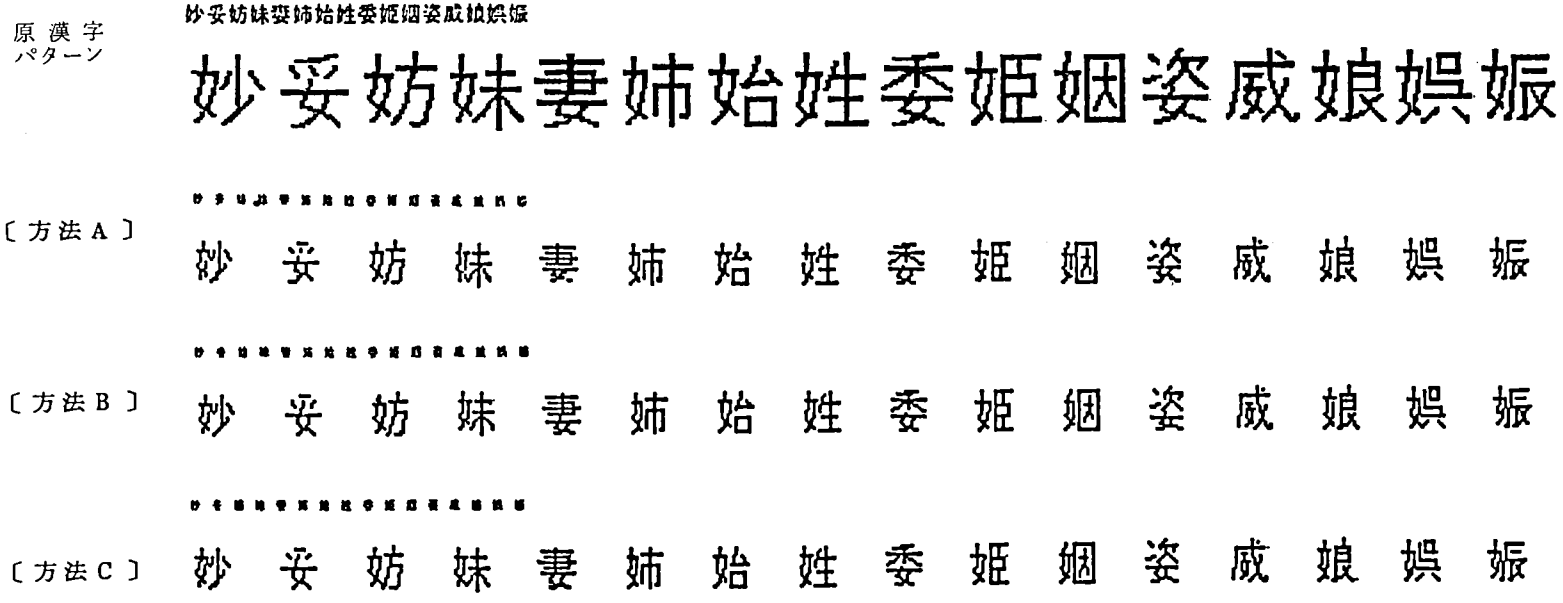


図 3.2 5　行列および画素選択情報付与方式による変換パターン例(3)

3. 4. 2　変換処理速度の改善

3. 3. 2 項で述べた線分の比例配置法における前処理として制御情報付与方式を見るとき，直線識別情報付与方式は線分の抽出処理まで前処理として実行しておくものであり，処理量としてはダイナミックステップ数にして約５０～６０％の処理量の削減をもたらす。

また，行列および画素選択情報付与方式では約７０～８０％の処理量の削減となる。

この結果，処理速度は２～５倍向上する。しかし，いずれにしろ，ソフトウェア処理によっては変換速度は高々１００字／秒程度に過ぎず，より高速化を達成するためにはハードウェア化が不可欠である。

したがって，制御情報付与方式は，その制御情報を用いることにより，いかにハードウェア化が容易になるかといった観点から評価することが必要となる。

ハードウェア化が困難な処理は行，列番号に関する写像関数の作成処理であるため，この処理を前処理として実行しておくことが望ましく，行列および画素選択情報付与方式が高速化のための制御情報付与方式としては優れているといえる。

以下，行列および画素選択情報付与方式について，ハードウェア化時の変換速度の目安を求める。

図 3. 2 6 に行列および画素選択情報付与方式のサイズ変換装置の構成ブロック図を示す。

制御情報に基づく行，列の選択は並列，直列変換レジスター（同図でＰ－Ｓと表示）とゲート群で構成される。

（ $m \times n$ ）の原漢字パターンを（Ｍ×Ｎ）の漢字パターンへと変換する場合の動作は以下の通りである。

漢字パターンメモリから列を単位（ m ビット長）として原漢字パターンを読み出しＰ－Ｓレジスタへ順次書き込む。この時，制御部は列選択情報を用いて，縮小変換の場合には同一のＰ－Ｓレジスタ内に２つの列パターンを重畳書き込みを行ない，拡大変換の場合には同一の列パターンを連続する２つのＰ－Ｓレジスタへ書き込むように制御することになる。

全ての列パターンのＰ－Ｓレジスタへの書き込みが終わった後，同一のシフトクロックを用いて N 個のＰ－Ｓレジスタから並列に変換漢字パターンの行パターン（ N ビット長）を順次読み出す。

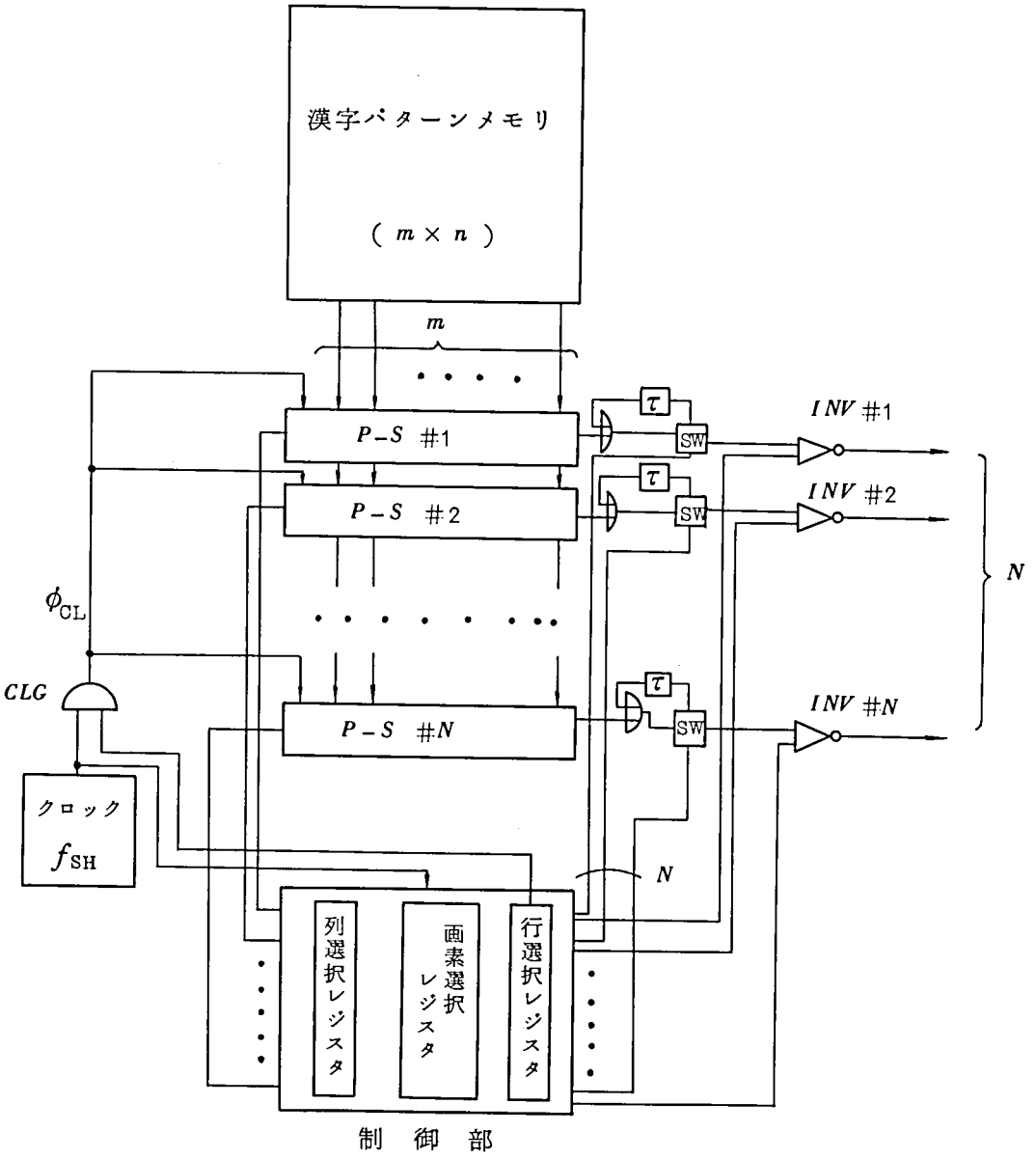


図 3.2 6　制御情報付与方式のサイズ変換装置構成ブロック図

この時，制御部は行選択情報に基づき，縮小変換の場合にはＰ－Ｓレジスタの出力をスイッチ回路（図でＳＷと表示）を制御し，必要な個数の行パターンの重畳削除を行ない拡大変換の場合にはクロックゲート（ＣＬＧと表示）を制御してシフトクロックを間引くことにより必要な個数の行パターンの反復挿入を行なう。

また，画素単位の修正は画素選択情報に基づいて，Ｐ－Ｓレジスタの出力信号をインバータを用いて反転することにより実現される。

以上の動作から，変換速度は基本的にＰ－Ｓレジスタのシフト速度で決まることに

なり，シフトクロックの周波数f_{SH}とすると，変換速度の目安として，縮小変換の場合はf_{SH}/m（字／秒），拡大変換の場合はf_{SH}/M（字／秒）が得られる。

今，シフトクロックの周波数f_{SH}を制御部の動作速度も考慮して実効上５００KHzとし，$m=32$，$M=48$とすれば，変換速度は縮小変換，拡大変換に対してそれぞれ，約１５,０００（字／秒）と１０,０００（字／秒）となる。

3.4.3 制御情報の作成

制御情報として行，列および画素の選択情報を付与する方式では，必要な変換サイズ毎に各文字について制御情報を作成することが必要であり，大きな作業量を伴う。そこで，制御情報作成の省力化が重要となる。

制御情報の自動作成法は処理時間さえ許せば，当然，独立したサイズ変換法として用いることが可能である。また，逆に，サイズ変換処理時間が長いため実時間でのサイズ変換法として用いられなかった方法も制御情報の自動作成法として用いることができる。

また一方，制御情報付与方式では変換後のサイズを特定し，制御情報の非実時間の作成が許容されることを充分利点として活かすためには，制御情報の作成を計算機との対話処理で行なうことが適している。会話処理で制御情報の作成を行なうことにより，変換漢字パターンの文字品質を予じめ確認し，必要なら修正を加えることが出来るという，実用面から見た大きな特長が生じる。

図3.27に制御情報の会話作成処理の流れを示す。また，図3.28に表示画面例を示している。

（３２×３２）の原漢字パターンをディスプレイ上に表示し，その左側と下側に表示された行および列の選択情報レジスタの内容をライトペンを用いて書きかえることにより行列選択情報を作成する。

この時，3.4.1項の方法Aとして，人間が最初から全く任意に行，列を選択することも，また，予じめ前述の自動変換アルゴリズムに従って計算機側で求めた行列選択情報を表示し，それを人間が修正することも可能である。

作成された行列選択情報に対応する変換漢字パターンは同一画面上に表示される。

行列選択情報が決定されると，それに対応する変換漢字パターンに対して画素単位の修正を行ない制御情報と変換漢字パターンが得られる。

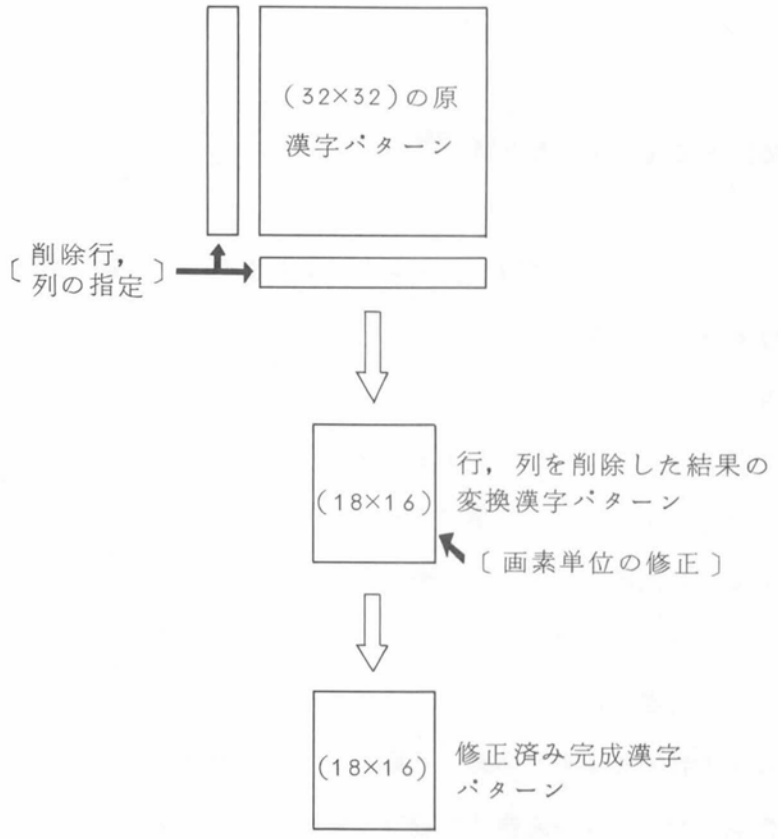


図 3.27　制御情報の会話作成手順

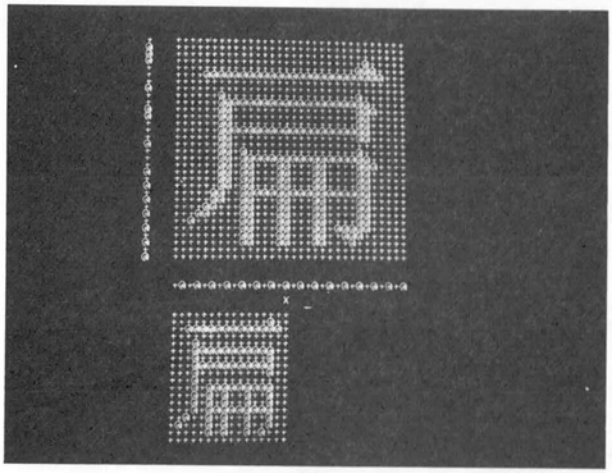


図 3.28　会話形制御情報作成画面

この方法で制御情報を作成した時の修正点数の分布は表3.5に示している。

（32×32）の原漢字パターンから（18×16）の変換漢字パターンを得るための制御情報の作成実験を約2900字に対して行なった結果，1字当りの平均作成

時間は約1.4分であった。

3.5 サイズ変換処理の適用領域と課題 (74)

サイズ変換法は変換可能な範囲，変換速度，変換文字品質，制御情報の有無などによって適用領域は異なってくる。

(1) 適用領域A

文書編集などへの適用であり，本文用の漢字パターンを原漢字パターンとしてもち，それ以外の標題用，脚注用などの漢字パターンをサイズ変換処理により発生させる。この適用領域では本文用の原漢字パターンが他のサイズの漢字パターンと比較してはるかに多く使用されるため，原漢字パターンの発生速度に比べて変換漢字パターンの発生速度は1～2桁程度遅くても許される。

他方，図表まで含めた編集を考えると必要となる変換漢字パターンのサイズは特定できないため制御情報を用いない変換法が望ましい。

また，記録として残せるだけの良好な変換文字品質でなければならない。

以上の条件から変換速度は遅いが制御情報を用いない線分の比例配置法が適用可能なサイズ変換法の候補となるが，広範囲な変換に対する変換文字品質の向上が必要である。

(2) 適用領域B

テレビ受像機やファクシミリ受信機など解像度の異なる装置へそれぞれ適切な大きさの漢字パターンをサイズ変換処理を用いて作成し供給する場合である。

この適用領域ではすべての大きさの漢字パターンを高速に発生してやることが必要となる。文字品質は，表示用の漢字パターンは記録用のそれに比べると若干の品質劣化は許容されると考えられるので，記録用の精細な漢字パターンからの高速な縮小サイズ変換技術が要求される。また，出力装置の種類が少なければ制御情報を用いた変換法も適用可能である。

以上の条件から適用可能なサイズ変換法としては制御情報付与報式が最も有望となる。

(3) 適用領域C

梱包ラベルや荷札などの宛名書き等のスタンドアロン形システムへの適用である。この適用領域ではきわめて広範囲での拡大変換が要求されるが，変換速度に対する条件は緩や

かである。変換文字品質については文字バランスの良さ，斜線部の滑らかさが要求される。

以上の条件から比例法が最も適当であるが斜線部の平滑化処理が必要である。

以上，まとめると，線分の比例配置法は制御情報を一切用いない特長があるが，変換速度が遅く，かつ，広範囲な変換に対して必ずしも満足すべき文字品質が得られない問題がある。

一方，制御情報付与方式は変換サイズ対応に制御情報をもつ必要があるが，充分実用に耐え得る文字品質が得られ，かつ，高速な変換が可能である。このため，すでに実用システム内に導入された例が見られる。

今後，本研究に残された課題としては，文字品質の定量的な評価法，制御情報付与方式の装置化それに文字品質が優れた完全自動のサイズ変換アルゴリズムの検討があげられる。

3.6 まとめ

漢字情報処理システムの社会への普及に伴なって，システムで要求される漢字パターンの種類は多くなるものと考えられる。

従来，画素形漢字パターン発生方式においては発生すべき漢字パターンは全て漢字パターン発生器内に予じめ蓄積しておくことが必要であり，複数種類の漢字パターンの発生要求に対しては，漢字パターン発生器が高価になると云う理由で応えることは出来なかった。

サイズを変換する研究としては本研究以前には，画素サイズを等価的に拡大する，整数倍の拡大処理が報告されている。

このような状況を背景として，本研究では拡大，縮小まで含めて，非整数倍のサイズ変換処理について考察した。

その結果

(1) サイズ変換を実現するために必要な画素の挿入，削除を字種対応に行なうという考え方を新たに導入した。

すなわち，整数倍の拡大処理も含めて，画素の座標の比例変換による従来のサイズ変換法では，挿入，削除される画素位置は漢字パターンの種類に依らず一定であったため，縮小サイズ変換では字画の欠落が生じ，拡大サイズ変換では線幅の不揃いが避けられず，

結果として，整数倍の拡大変換のみが可能であった。

新たに挿入，削除画素を漢字パターン対応に選択する考え方を導入することにより，拡大，縮小を含めて非整数倍のサイズ変換を可能とした。

(2) 漢字パターン対応に挿入，削除画素を選択する考え方を導入するに伴なって，サイズ変換に対する要求条件と画素の選択基準を新たに提示した。

(3) 前記の条件を基に，行，列を単位として画素を挿入，削除することにより，文字品質の主要な支配要因である縦横直線の線幅と文字バランスの両者が制御可能な全自動のサイズ変換アルゴリズム（線分の比例配置法）を提案した。

(4) 漢字パターン対応に挿入，削除する画素を行，列を単位として選択する考え方の具体的な手法として，行列の選択情報を制御情報として付与することにより，高速変換が可能で，かつ，変換文字品質も良好な制御情報付与方式を実用可能な手法として提案した。

今後に残された課題として

(1) 文字品質の定量的な評価法の確立

(2) 変換文字品質に優れた全自動変換アルゴリズムの開発

(3) 各変換アルゴリズムのハードウェア化

があげられる。

第4章　書体変換処理

我が国では代表的な書体として明朝体とゴシック体が用いられている。これらの2つの書体は，例えば，見い出しにはゴシック体，本文には明朝体というように用途により使い分けられている。

このような，既存の活字文化の中に漢字情報システムが本格的に受け入れられるためには，他の書体がないため，1つの書体で代用するということは許されない。

そこで，1つの字種に対して複数書体の漢字パターンを持とうとすれば，漢字パターン発生器が高価になるという問題が生じる。

この問題を解決する技術としては異なる書体間での書体変換処理が考えられるが，従来画素形漢字パターンの書体変換処理に関する研究は見当らない。

そこで，本研究では代表的な書体である明朝体とゴシック体をとり上げ，両書体間の書体変換の可能性について考察する。

4.1　まえがき

3章で考察したサイズ変換処理は基本的にはパターンの相似変換であったのに対して，書体変換は形状の変化を伴なうため，より高度な処理が要求される。

書体変換処理の研究は本研究以前には見当らないが，本研究以降，幾つかの研究が報告されている。(42)

書体変換処理は文字のストロークの形状変化を伴なう処理であるため，漢字パターンの表現形式としてはストローク形式が適している。この利点を活かして，基本的な漢字パターンの表現にはストローク形式を用い，それ以外にそのストローク情報に従って動く“筆さき”の形状を制御する情報をもつことにより，発生させる漢字パターンの書体に自由度を持たせる研究が見られる。(44)(75)(76)

画素形式で表現された漢字パターンの書体変換の考え方としては次の2通りがある。

① ストローク抽出方式

② 差分パターン処理方式(40)

ストローク抽出方式は，画素形漢字パターンからストローク情報を抽出し，ストローク情報を媒介にして書体変換を行なう方式であり，本研究ではこの方式をとっている。この場合にはストロークの抽出処理が重要な処理となる。

一方，差分パターン処理方式は，両書体間の差分パターンを別途，書体変換のための付属情報として持つ方式であり，同一字種であれば，異なる書体間の差分パターンがもつ情報量は，それぞれの書体の漢字パターンを独立して表現するのに必要な情報量より少なくて済むという考え方に基づいている。(39)

こゝでは，以下，（32×32）の漢字パターンをとりあげ，ストローク抽出方式による画素形漢字パターンの書体変換処理について述べる。

4.2 基本的考え方と処理の流れ

ストローク抽出による書体変換法ではストロークの抽出が必要である。画素形文字パターンからのストロークの抽出法の研究は印刷文字の自動認識の分野でストローク・アナリシス法として研究例が見られるが，(77) ストロークの完全な抽出は現状では不可能といえる。

しかし，書体変換の目的と漢字認識の目的との違いにより，ストロークの抽出条件は異なる。

書体変換の最終目的は両書体間のパターン形状の差異を制御できれば良く，それに必要なストローク情報の抽出さえ出来れば良い。(78)

(1) 明朝体，ゴシック体の差

図 4.1 にサイズ（32×32）の明朝体とゴシック体の漢字パターンとその差分パターンの例を示す。

この例から分かるように，両書体間の差異は次の3点となる。

① 縦，横直線の線幅比 r（明朝体は $r=2$，ゴシック体は $r=1$）

② 縦，横直線の線端に位置する“ウロコ”等の線端飾りの有無

③ 斜線，曲線の形状

これらの差異のうち③の斜線，曲線の形状は同じ明朝体でもデザインによって異なり，一意的に表現形式が定められていない。したがって，こゝでは①，②の差異の制御を対象として以下の考察を行なう。

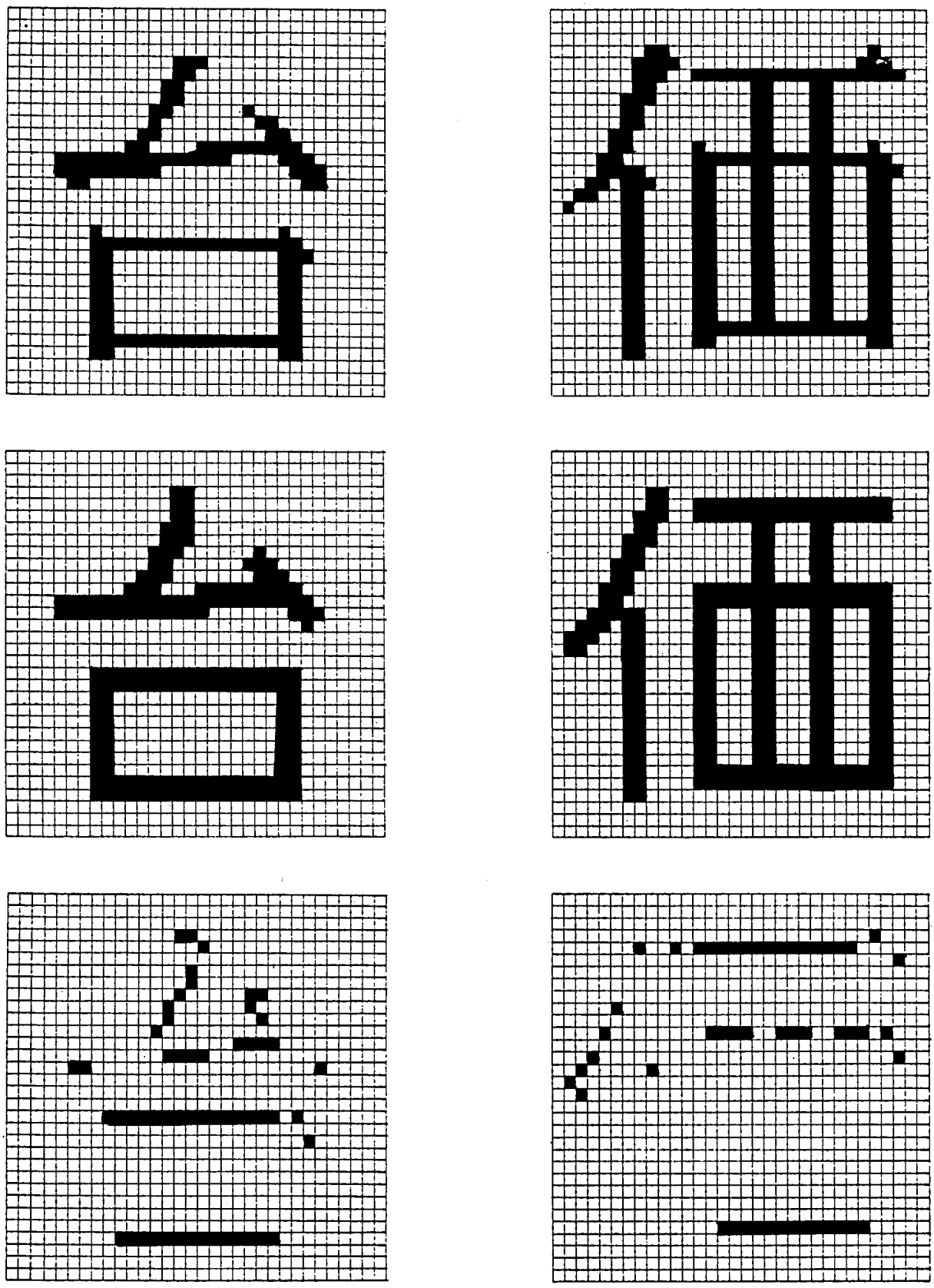


図 4. 1　明朝体，ゴシック体および差分パターンの例(1)

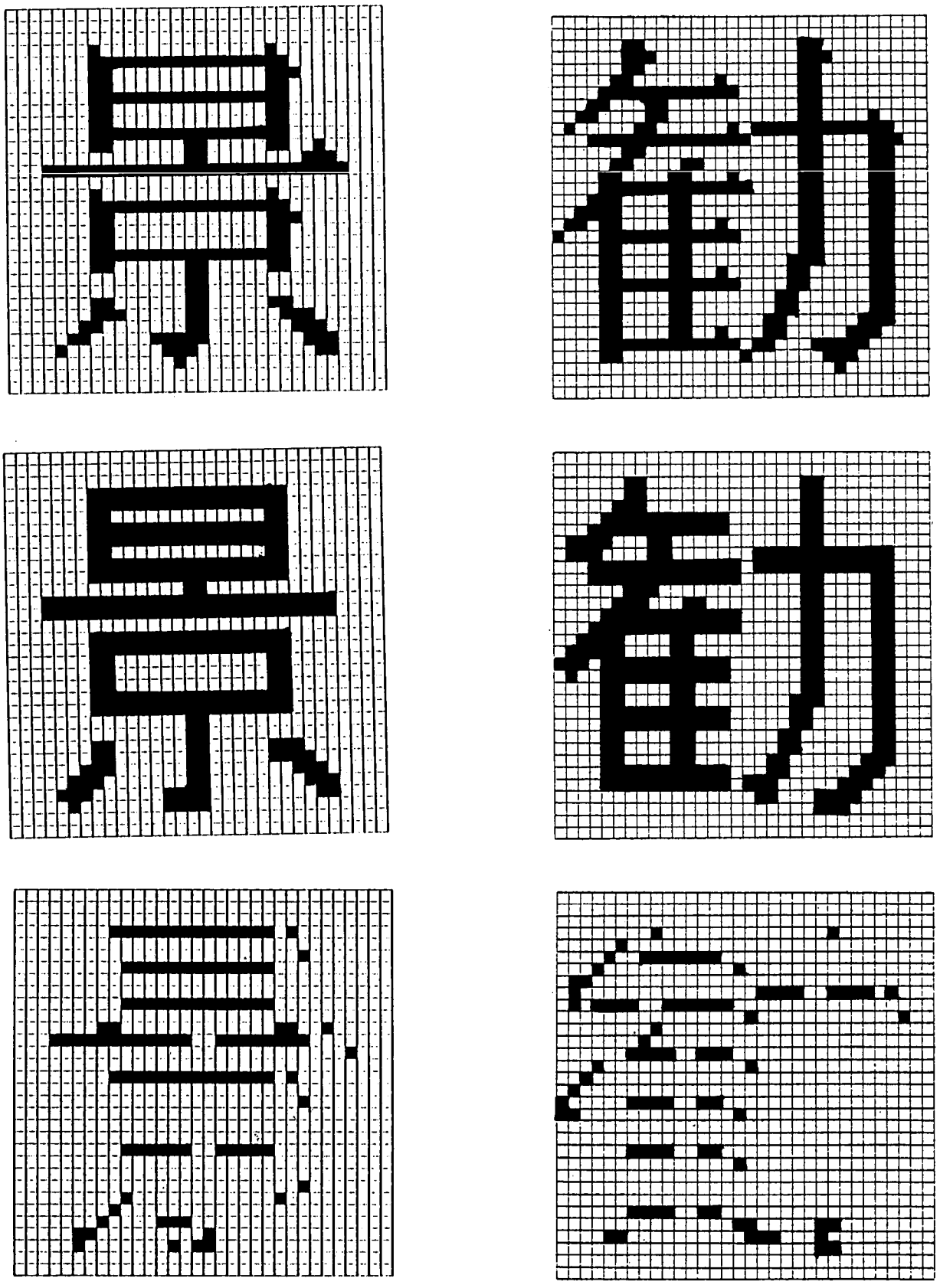


図 4. 1　明朝体，ゴシック体および差分パターンの例(2)

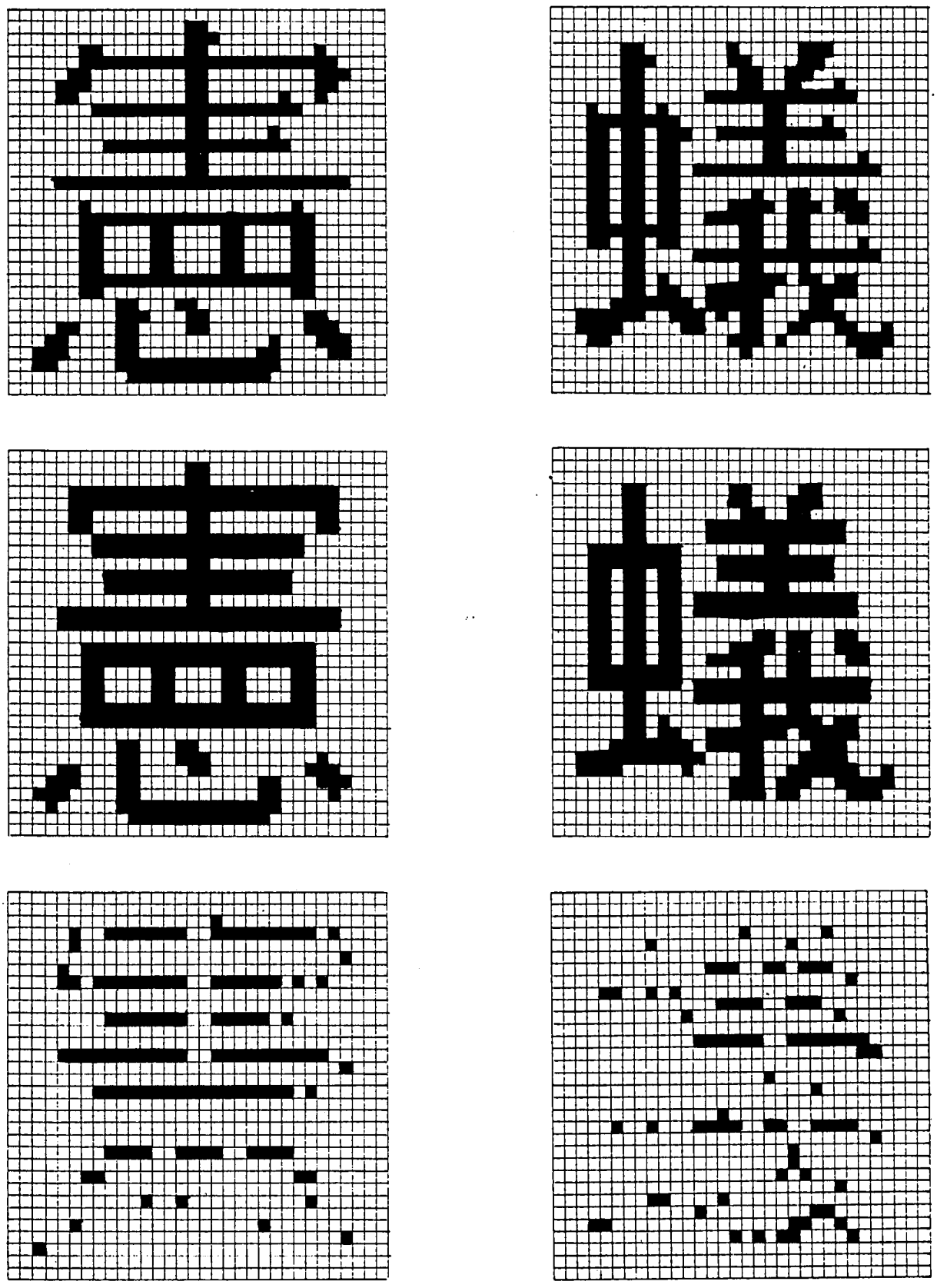


図 4. 1　明朝体，ゴシック体および差分パターンの例(3)

図 4.2 には以下の書体変換処理で考察の対象とする線端飾りの種類を示す。これらの線端飾りの形状は漢字パターンのサイズによって異なるが，ここでは（32×32）の明朝体漢字パターンに現われる代表的なものとして選定した。

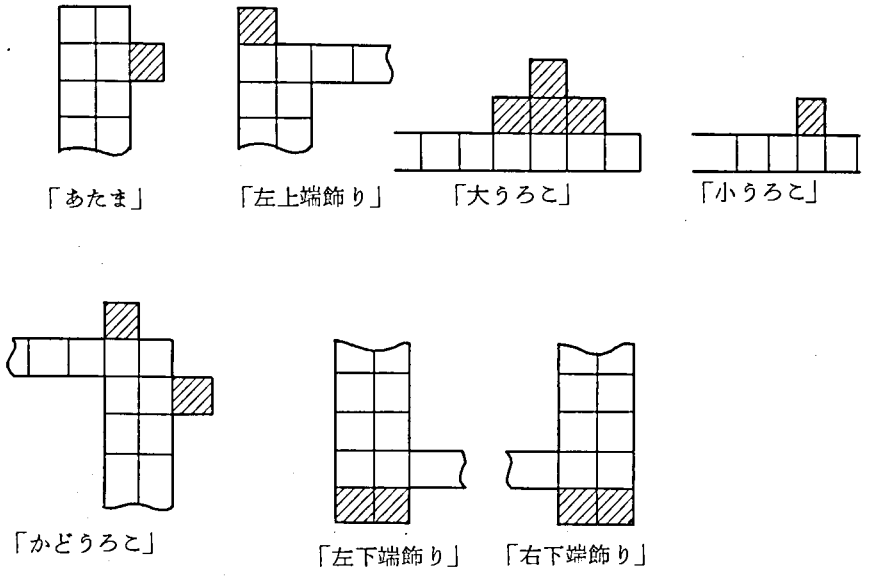


図 4.2　考察の対象とした線端飾りの種類

(2)　基本的な考え方

縦，横直線の線幅比と線端飾りの有無を制御の対象とする時，書体間の差異は縦横直線のみの形状の変化としてとらえることができる。

そこで，漢字パターンを構成する縦横直線と斜線曲線の分離処理を基本的な処理方針とした。

図 4.3 に書体変換処理の流れを示す。

処理は変換方向，すなわち明朝体からゴシック体への変換とゴシック体から明朝体への変換とによって内容が異なる。

明朝体からゴシック体への変換では，線端飾りを縦横直線から切り離すため，漢字パターンを縦横直線のみから成る直線部と斜線，曲線それに線端飾りから成る斜線曲線部へと一旦分離した後，再合成するという処理過程を踏む。

いずれの変換においても，次の 3 つの処理内容から成る。

①　縦横直線の抽出

②　横直線の線幅の変更

③　線端飾りの制御

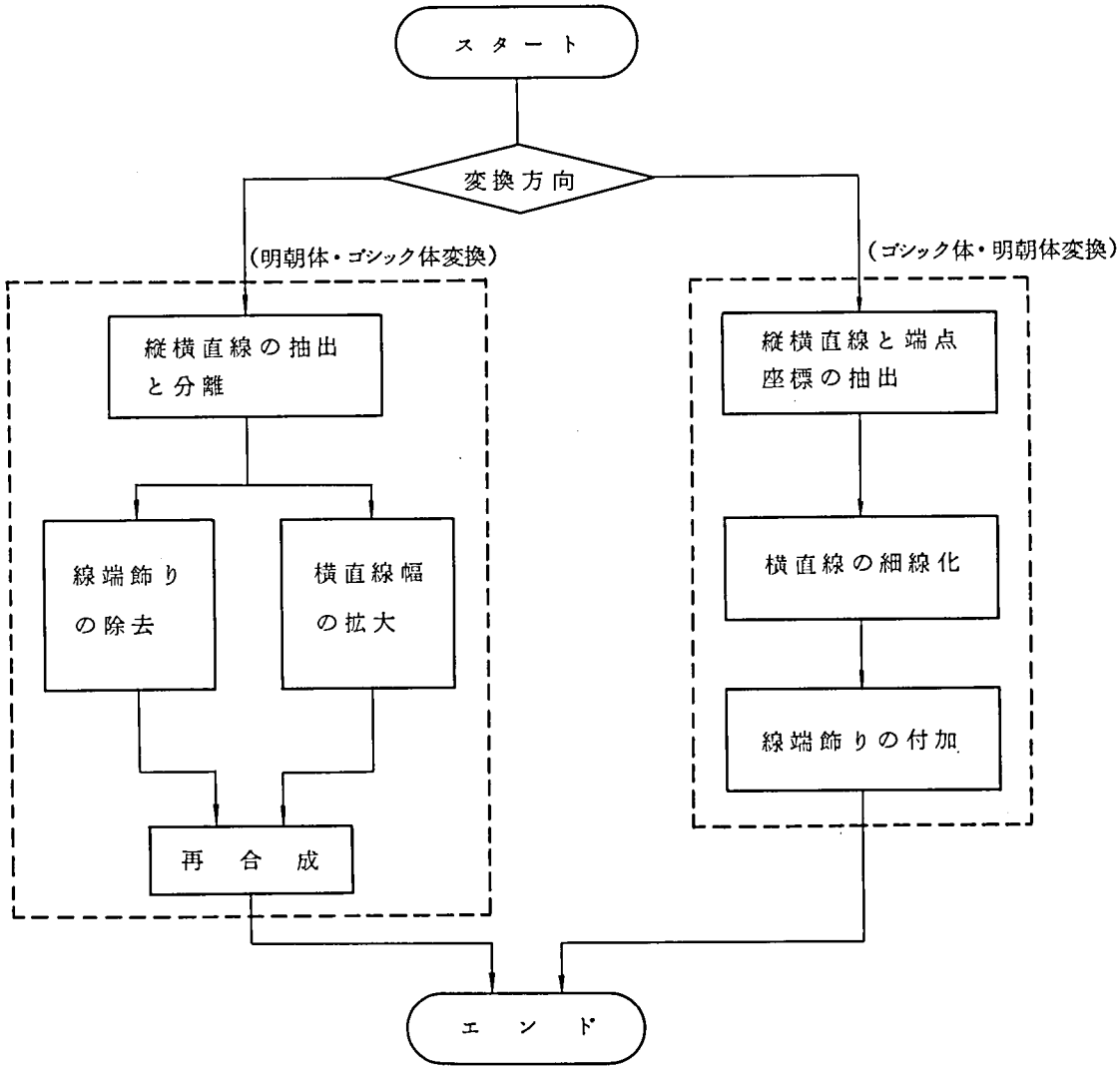


図 4. 3　書体変換処理の流れ

4.3　明朝体からゴシック体への変換

明朝体からゴシック体への変換では，分離された縦横直線のみからなるパターン(以後，縦横直線パターンとよぶ）に対しては横直線の線幅の太め処理，斜線曲線のみからなるパターン（斜線曲線パターンとよぶ）に対しては線端飾りの除去処理を行なう。

4.3.1　縦横直線の抽出と分離

線端飾りは縦直線と横直線の位置関係によって，その付加位置と種類が決まるが，

明朝体からゴシック体への変換では線端飾りを除去するだけで充分であり，その飾りの種類まで識別する必要はない。

したがって，線幅の拡大処理が必要となる横直線のみを抽出すればよい。

付録 4. 1 に示す 40 字種の（32×32）の明朝体漢字パターンについて，線幅の変更と線端飾りの消去が必要となる横直線の長さの分布を調べた。その結果を表 4. 1 に示す。

表 4. 1　書体変換処理で抽出対象となる縦横直線の長さ分布

（試料数；40字）

長　さ	割　合		長　さ	割　合	
	横直線 *1)	縦直線 *2)		横直線 *1)	縦直線 *2)
4	0	0.3	17	6.1	3.5
5	2.2	4.5	18	5.0	2.9
6	2.8	5.1	19	2.2	3.5
7	9.4	10.6	20	1.7	1.6
8	6.1	6.4	21	1.7	1.9
9	5.0	4.2	22	2.2	1.6
10	10.6	3.9	23	1.1	2.9
11	3.9	7.4	24	0.6	2.3
12	9.4	7.1	25	2.2	2.6
13	5.6	5.8	26	1.7	3.2
14	7.2	4.8	27	1.7	2.3
15	6.1	4.2	28	0.6	1.6
16	4.4	5.8	29	0.6	0

*1）……（32×32）明朝体漢字パターン

*2）……（32×32）ゴシック体漢字パターン

この結果，長さ5以上の横直線を抽出の対象とすれば良いことが分かる。

そこで，漢字パターンを横方向に走査し，長さが5以上の横直線について両端の画素位置（I，Js），（I，Je）を求め横直線を抽出する。

なお，始点，終点がそれぞれ（I，Js），（I，Je）の直線を

$$\{(I, Js), (I, Je)\} \qquad (4.1)$$

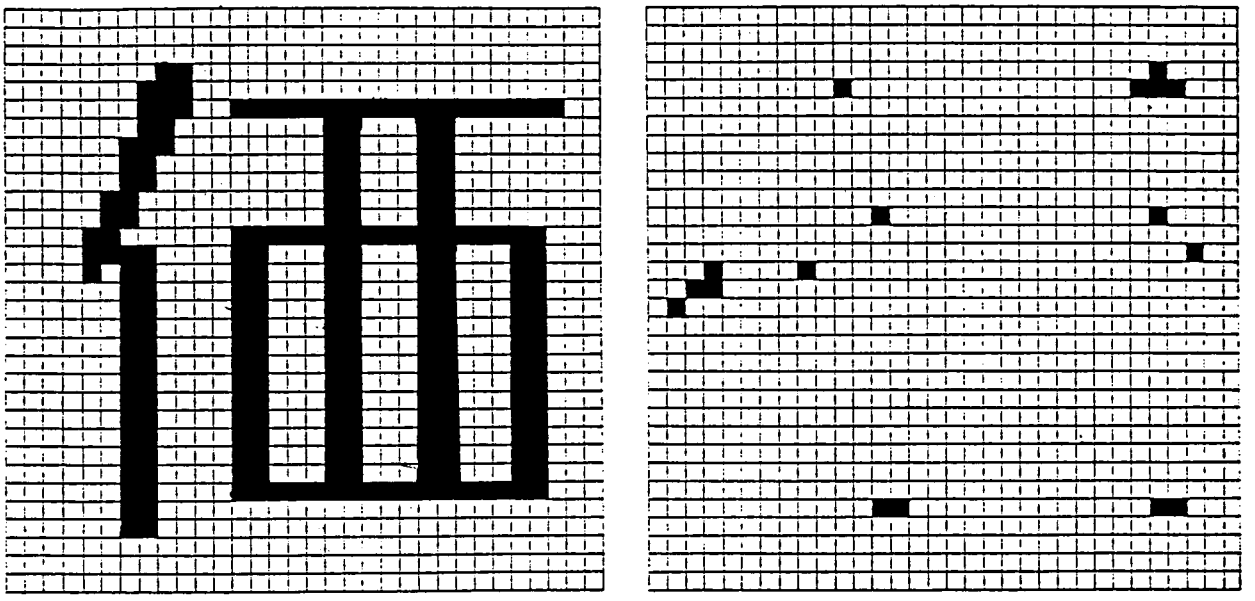
と表わす。座標系は図 4. 5 に示す。

一方，縦直線は横直線を抽出分離した後では短かく分断され，斜線，曲線そして線端飾りと混在することになる。

この漢字パターンから横直線のみを分離して残ったパターンでは，線端飾りはなお，分断された縦直線の 1 部と連結されて存在している。そこで，線端飾りを縦直線から完全に切り離すため，明らかに線端飾りの構成要素でない黒画素を除去する。すなわち，線端飾りの構成黒画素は縦方向に 3 画素以上連続しないことから，縦方向に 3 画素以上連続する黒画素を分離する。但し，縦方向の線分については座標情報を抽出する必要は無い。

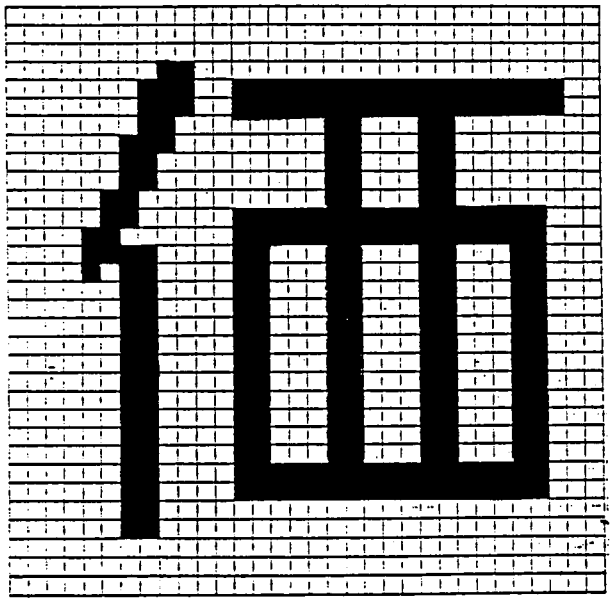
以上，横方向に 5 画素以上，縦方向に 3 画素以上連続する黒画素から成るパターンを縦横直線部，原漢字パターンから縦横直線部を除いた残りのパターンを斜線曲線部とする。

縦横直線部と斜線曲線部の例を図 4. 4 (a), (b)に示す。

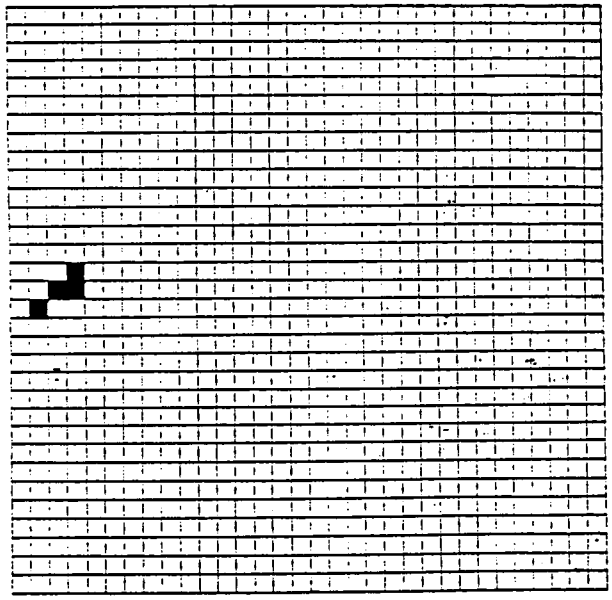


(a) 縦横直線部の例　　(b) 斜線・曲線部の例

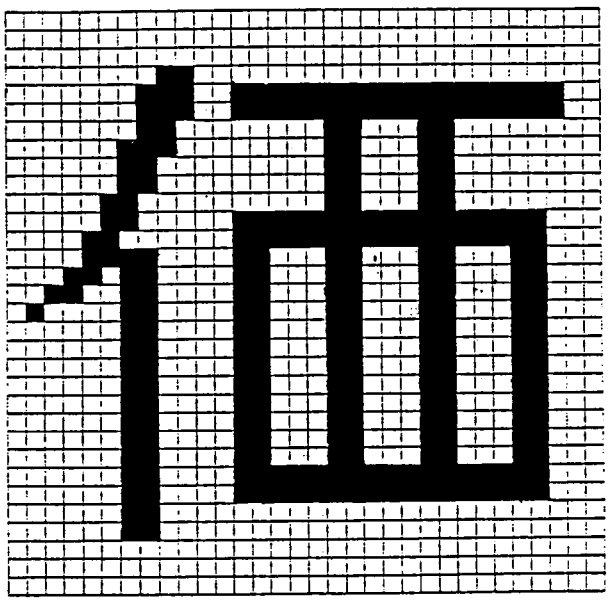
図 4. 4　明朝体からゴシック体への書体変換処理過程のパターン例　（その 1 ）



(c) 横直線幅を拡大した縦横直線部



(d) 線端飾りを除去した斜線曲線部



(e) 縦横直線部と斜線曲線部の再合成
により得られた変換ゴシック体

図 4.4　明朝体からゴジック体への書体変換処理過程
のパターン例　（その２）

4. 3. 2　横直線幅の拡大処理

縦横直線の分離処理によって抽出された横直線

$$\{(I, Js), (I, Je)\} \tag{4.2}$$

の線幅を以下の条件で拡大する。

①　拡大方向は上側とする。

すなわち，新たに横直線

$$\{(I-1, Js), (I-1, Je)\} \tag{4.3}$$

を発生する。

ただし

②　線幅を拡大することにより，上位の横直線との間の間隙が無くなる場合には線幅は拡大しない。

③　下側に隣接して横直線が存在する場合には，接する範囲については線幅は拡大しない。

すなわち，式（4.2）の横直線の下側に，横直線

$$\{(I+1, Js'), (I+1, Je')\} \tag{4.4}$$

が存在する時，

ⓐ　$Js < Js' \leqq Je < Je'$ なら

横直線：$\{(I-1, Js), (I-1, Js')\}$ (4.5)

ⓑ　$Js < Js' < Je' < Je$ なら

横直線：$\{(I-1, Js), (I-1, Js')\}$ (4.6)

横直線：$\{(I-1, Je'), (I-1, Je)\}$ (4.7)

ⓒ　$Js' < Js \leqq Je' < Je$ なら

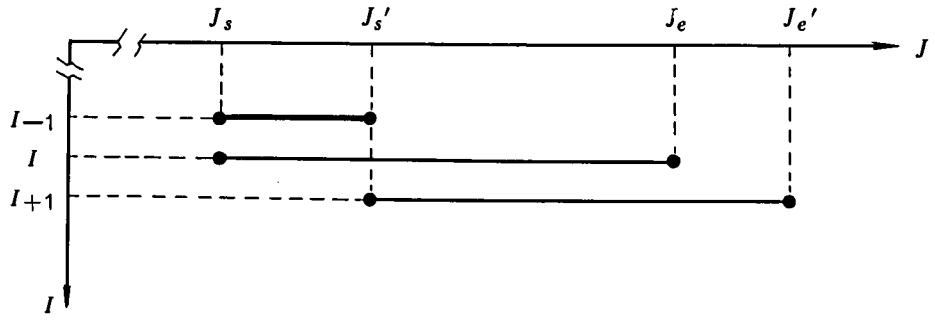
横直線：$\{(I-1, Je'), (I-1, Je)\}$ (4.8)

をそれぞれ新たに発生する。これらの状況を図 4.5 に示す。

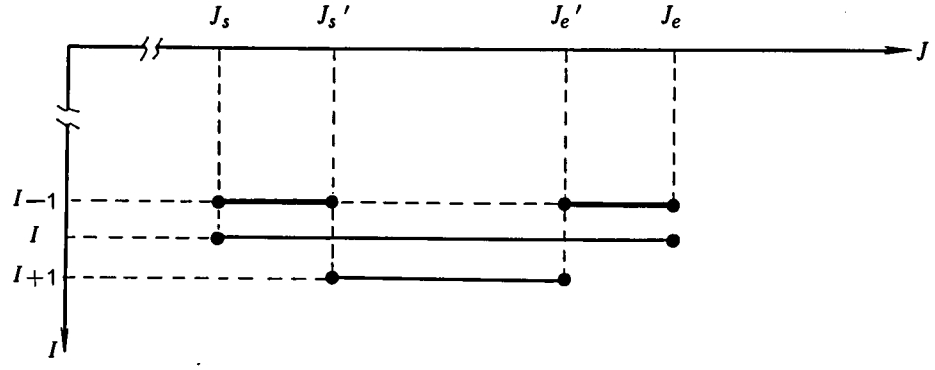
②の線幅の拡大方向としては上側，下側の 2 通りがあるが，拡大方向を上側としたのは横直線に付随した線端飾りの多くは上側に位置しており，その結果，字形の構成上，横直線の上側が線幅拡大のための余地が広いからである。

又，③は横直線の線幅を 2 画素に揃えるためである。

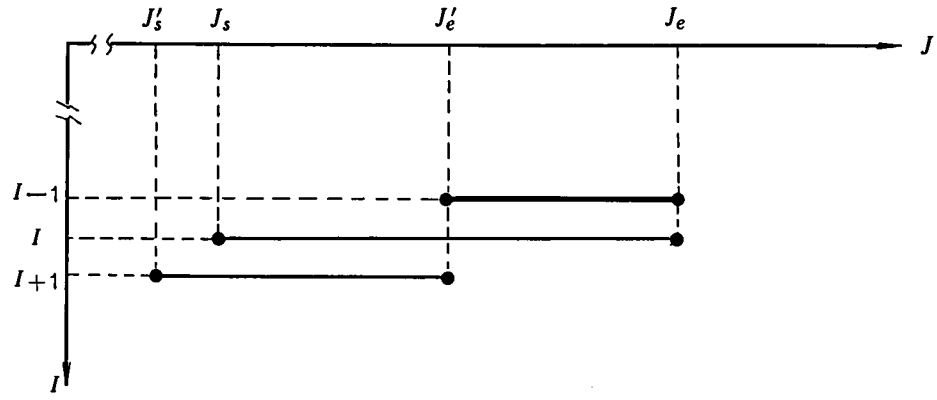
横直線の線幅を拡大した縦横直線部の例を図 4.4 (c)に示す。



(a)　$J_S < J_S' \leqq J_e < J_e'$ の場合



(b)　$J_S < J_S' < J_e' < J_e$ の場合



(c)　$J_S' < J_S \leqq J_e' < J_e$ の場合

図4.5　横線幅の拡大

4.3.3　線端飾りの除去

斜線，曲線部には斜線，曲線の一部と線端飾りが含まれる。

図4.2に示す線端飾りの形状から分かるように，斜線，曲線の一部と線端飾りを分離する最も有効なパラメータは連結する黒画素の個数である。すなわち，縦横直線から分離された線端飾りの黒画素の連結個数は4連結状態で高々4であり，それ以上連

結した黒画素は無条件で斜線，曲線の一部であると判定することができる。

そこで，斜線曲線パターンから以下の条件により線端飾りを除去する。

① 4連結状態で，連結個数が2以下の黒画素は無条件に除去する。

② 4連結状態で，連結個数が4の黒画素の中で，黒画素の配置が図4.2の「大ウロコ」を形成していれば，それらの黒画素を除去する。

以上の条件から，具体的には，図4.6(a)に示す（3×4）画素から成るウィンドウで，斜線曲線パターンを左上端から，左から右へ，上から下へと走査し，次式で定義するウィンドウ値を算出することにより，線端飾りを抽出し除去する。

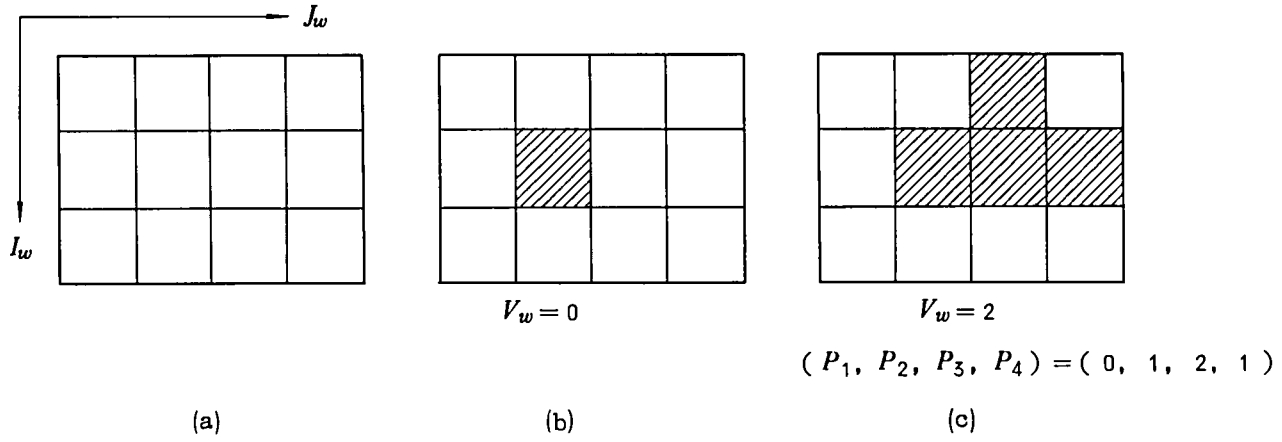


図 4.6　線端飾り除去処理用ウィンドウと構成画素の例

$$V_w = \sum_{Jw=1}^{4} \{W(1, Jw) + W(3, Jw)\} + W(2,1) + W(2,4) \qquad (4.9)$$

こゝで$W(Iw, Jw)$，($Iw = 1, 2, 3$, $Jw = 1, 2, 3, 4$)は，各ウィンドウ位置で切り出した時の斜線，曲線部の画素状態を示し，黒画素であれば1，白画素であれば0とする。

式（4.9）で与えられる値V_wはウィンドウの外周上の黒画素の個数を示すことになる。

そこで

(i) $V_w = 0$の場合

$$\left.\begin{aligned} W(2,2) &= 0 \\ W(2,3) &= 0 \end{aligned}\right\} \qquad (4.10)$$

とする。

(ii) $V_w \leqq 3$ の場合

$$P_J = \sum_{Iw=1}^{3} W(Iw, J) \qquad (4.11)$$

$$J = 1, 2, 3, 4$$

を求め，集合 (P_1, P_2, P_3, P_4) と集合 $(0, 1, 2, 1)$ の間に巡回置換が存在すれば，

$$W(Iw, Jw) = 0 \qquad (4.12)$$

$$\begin{pmatrix} Iw = 1, 2, 3 \\ Jw = 1, 2, 3, 4 \end{pmatrix}$$

とする。

図 4.6 (b)，(c)に，それぞれ上記(i)，(ii)の場合のウィンドウの画素構成例を示す。

図 4.4 (d)に飾りを除去した斜線曲線部の例を示す。

4.3.4 再 合 成

横直線の線幅を拡大した縦横直線部と線端飾りを除去した斜線曲線部を合成することにより，ゴシック体を得る。得られたゴシック体の例を図 4.4 (e)に示す。

図 4.7 に変換処理によって得られたゴシック体漢字パターンの例を示す。

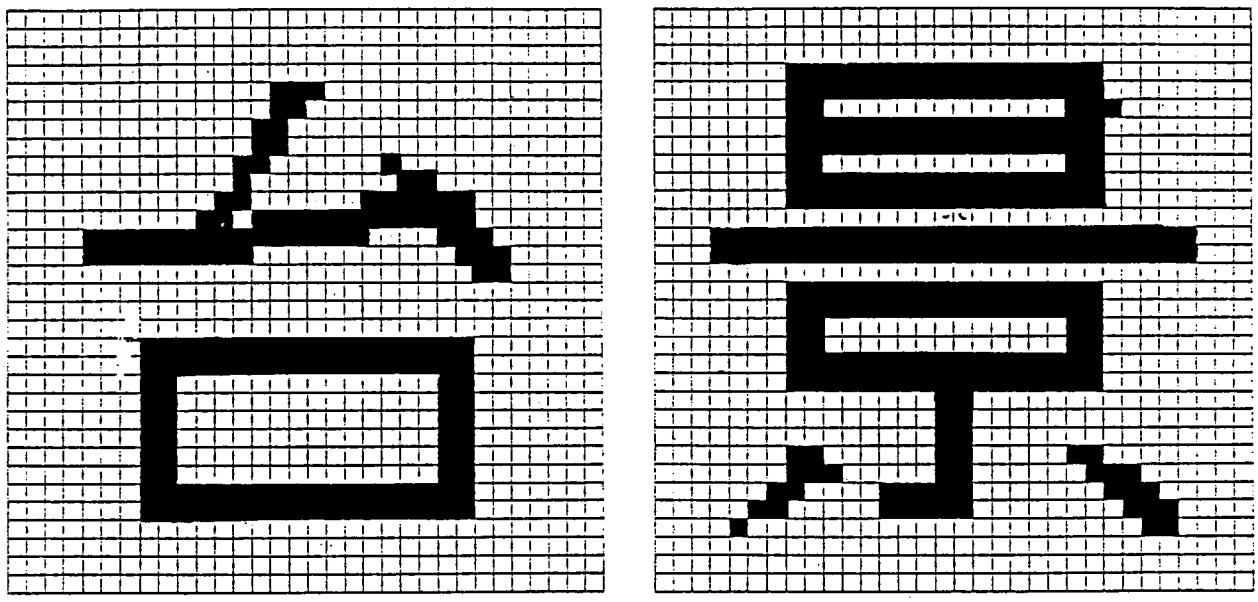


図 4.7　変換ゴシック体の例（その１）

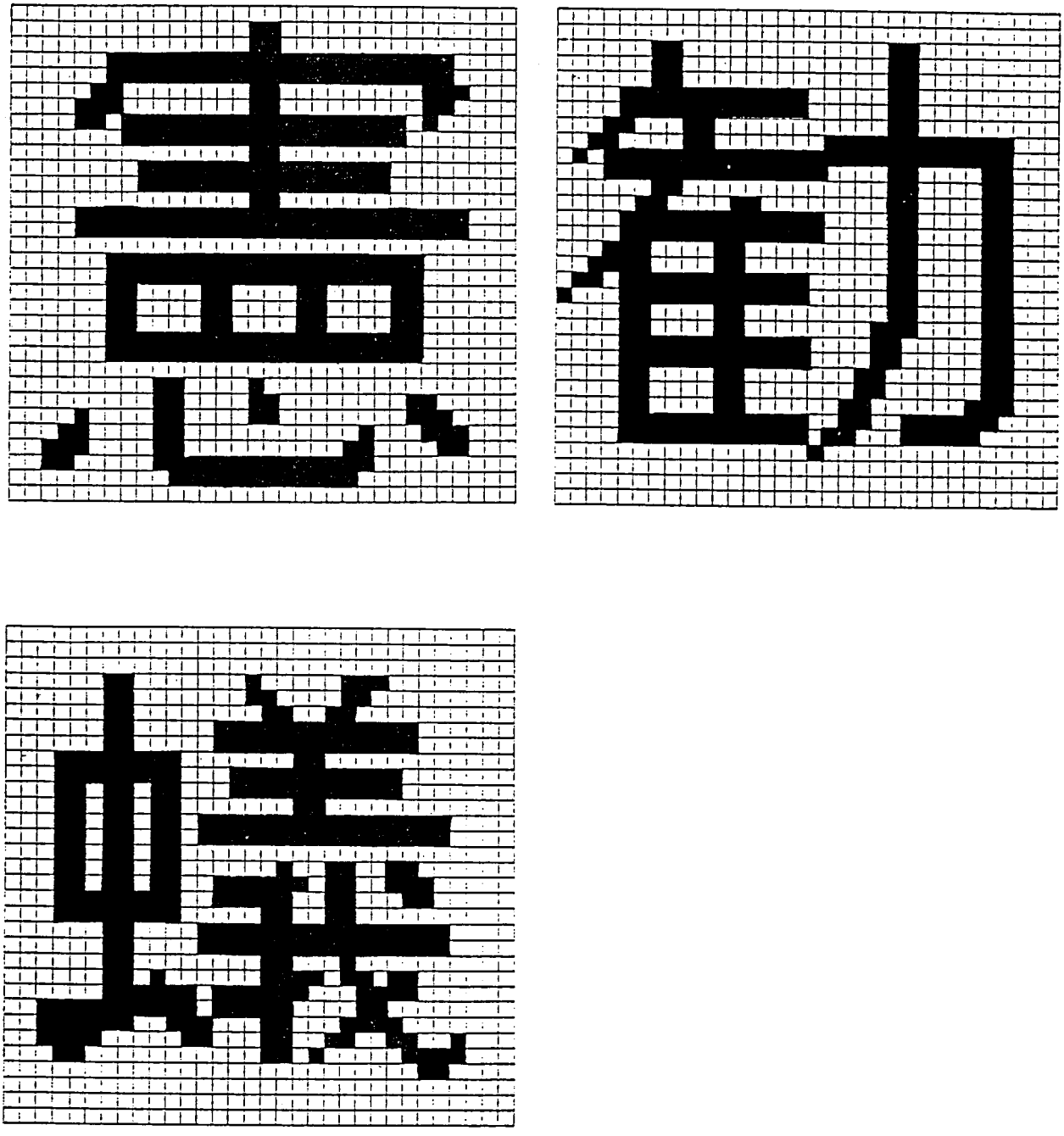


図 4.7　変換ゴシック体の例（その 2 ）

4.4 ゴシック体から明朝体への変換[79]

ゴシック体から明朝体への変換においては原漢字パターン内に存在しない線端飾りを新たに付加しなければならない。付加すべき線端飾りの種類と位置を決定するために，横直線に加えて縦直線情報も抽出する必要がある。

また，横直線の線幅の細線化により，縦直線と斜線，曲線に生じる断点を接続しなければならない。

以上の2点が明朝体からゴシック体への変換との主要な相違点である。

4.4.1 縦横直線の抽出

横直線との位置関係において，線端飾りの制御に関与する縦直線の長さの分布の調査結果を表4.1に示している。調査対象は付録4.1に示す40字種の（32×32）のゴシック体漢字パターンである。

この結果，長さ4以上の縦直線を抽出対象とすれば良いことが分かる。横直線に対しては4.3節と同じく長さ5以上とする。

ゴシック体から明朝体への変換では縦横直線と斜線曲線の分離は行なう必要が無い。（32×32）のゴシック体漢字パターンでは縦，横直線とも線幅が2画素であり，かつ，その直線を構成する黒ランの長さが異なる場合が多く，完全な縦横直線として両端を一意的に求めることは多くの場合困難となる。

しかし，ゴシック体から明朝体への書体変換の目的に対しては，

(i) 縦直線，斜線曲線は変形しない。

(ii) 線端飾りは縦横直線の斜線曲線と接しない側の端点に位置する。

の2点を考慮することにより，両端をもった完全な縦横直線情報の抽出は不要であり，以下の条件で必要な縦横直線の端点座標を抽出することができる。

すなわち，項目(ii)では，線端飾りの付加対象の縦直線としてはそれを構成する2本の隣接する縦黒ランについて，2つの端点のうち少なくとも一方の縦軸座標値（I座標値）が一致するものを，同じく，線端飾りの付加対象の横直線は2本の横黒ランの少なくとも一方の端点の横座標値（J座標値）が一致するものを抽出すれば充分であることを示している。

このことから，具体的には以下の条件で縦横直線情報（端点座標値）を求める。

〔横直線〕

隣接する2本の横方向の黒ランで長さが5以上のものである。

〔縦直線〕

隣接する2本の縦方向の黒ランで長さが4以上のものである。

すなわち，横直線としては次の2本の横黒ラン

$$\begin{cases} LH1\ ;\ \{(I_h-1, J_{s1}),\ (I_h-1, J_{e1})\} & (4.13) \\ LH2\ ;\ \{(I_h,\ J_{s2}),\ (I_h,\ J_{e2})\} & (4.14) \end{cases}$$

但し，

$$J_{e1} - J_{s1} \geqq 5$$

$$J_{e2} - J_{s2} \geqq 5$$

が，また，縦直線としては次の2本の縦黒ラン

$$\begin{cases} LV1\ ;\ \{(I_{s1},\ J_v-1),\ (I_{e1},\ J_v-1)\} & (4.15) \\ LV2\ ;\ \{(I_{s2},\ J_v),\ (I_{e2},\ J_v)\} & (4.16) \end{cases}$$

但し，

$$I_{e1} - I_{s1} \geqq 4$$

$$I_{e2} - I_{s2} \geqq 4$$

がそれぞれ抽出される。

但し，この段階では，それぞれ2本の黒ランがもつ2個の始点，終点の座標値を一意的に決定する必要は無い。

4.4.2 横直線の細線化

前項で抽出した縦横直線を用いて横直線の細線化とそれにともなう，縦直線の始点補正を行なう。

横直線の細線化を行なう場合，次のような問題点が生じる。

① 細線化の方向

② 横直線と交叉する縦直線，斜線，曲線の連続性の保存

③ 縦直線の始点位置補正

(i) 細線化の方向

横直線を形成する2本の黒ランのいずれの位置に細線化された直線を設定するかであるが，細線化直線の上側に線端飾りを付加するため，細線化直線を下側の黒ランの

位置とする。

但し，細線化の範囲は2本の横黒ランが互いに隣接する範囲とする。

すなわち，式（4.13），（4.14）で表わされる黒ランの構成画素を$a(I, J)$とするとき

$$\max(J_{s1}, J_{s2}) < J_1 < \min(J_{e1}, J_{e2}) \tag{4.17}$$

の範囲に対して

$$a(I_h - 1, J_1) = 0 \tag{4.18}$$

とする。

なお，細線化に対して次の除外規定を付ける。

〔細線化の除外規定〕

「隣接する長さ5以上の横黒ランであっても，その始点，終点の少なくともいずれか片側で，両黒ランの横座標値の差が所定の閾値以上であれば，細線化処理は行なわない。」

この細線化の除外規定は緩やかな勾配の斜線，曲線の線幅を保存するために設けたものである。

横座標値の差に対する閾値は，5章で述べる検討結果を基に，（32×32）の漢字パターンでは3ないし4が適当となる。

(ii) 交叉線分の連続性の保持

横直線の細線化によって生じる1画素幅の空隙は，横直線を構成していた2本の黒ランの上位黒ランと上側で接していた画素で補間することにより，交叉線分の縦方向への連続性を保持する。

すなわち，式（4.13），（4.14）の黒ランとして抽出された横直線に対して，

$$a(I_h - 1, J) = a(I_h - 2, J) \tag{4.19}$$

$$\max(J_{s1}, J_{s2}) \leqq J \leqq \min(J_{e1}, J_{e2}) \tag{4.20}$$

とする。

(iii) 縦直線の始点位置補正

横直線の細線化の方向を下側としたことにより，横直線上に始点をもつ縦直線の始点を細線化した画素分だけ下位へずらす。

すなわち，

$$I_{s1} = I_{s2} = I_h - 1 \tag{4.21}$$

かつ

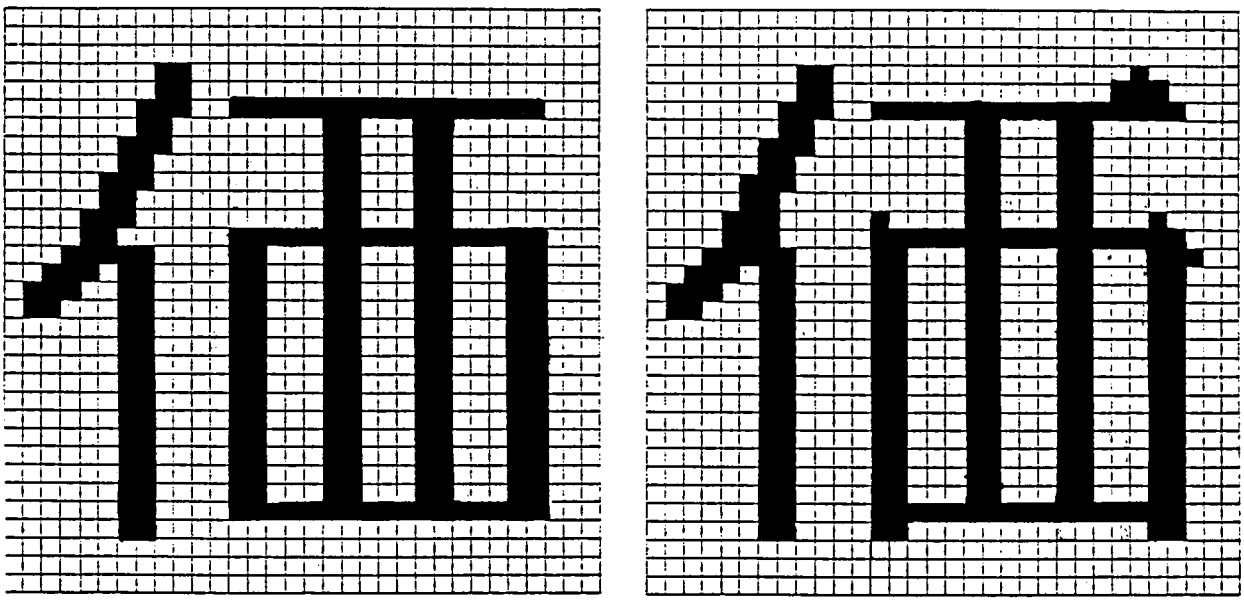
$$J_{s1}+1 \leqq J_v \leqq J_{e1} \qquad (4.22)$$

なら，

$$I_{s1}=I_{s2}=I_h \qquad (4.23)$$

とする。

以上の処理により，横直線が1画素幅に細線化された漢字パターンが得られる。この例を図4.8 (a) に示す。



(a) 横直線を細線化されたゴシック体漢字パターン　　(b) 変換明朝体の例

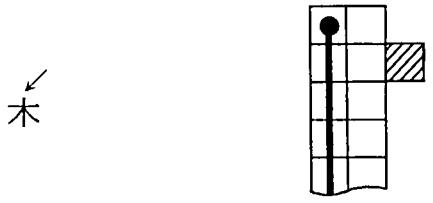
図4.8　ゴシック体から明朝体への変換処理過程のパターン例

4.4.3　線端飾りの付加

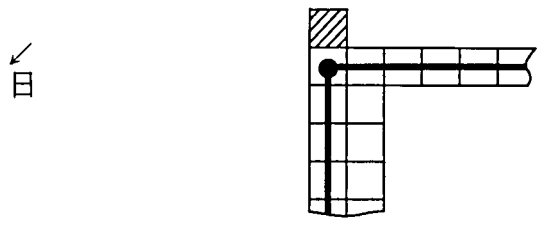
図4.2に示した明朝体における線端飾りの種類とその付加される位置は縦，横直線の線端の状態により一意的に決めることができる。

そこで，ここでは，4.4.1項で抽出した縦横直線を用いてその線端の状態を識別することにより，線端飾りの種類と位置の識別処理について考察する。

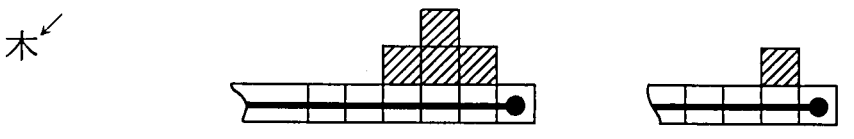
図4.9に縦横直線の端点の位置と線端飾りの種類，およびその構成画素の位置関係



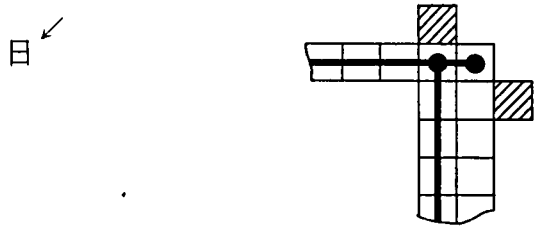
(a)　「あたま」



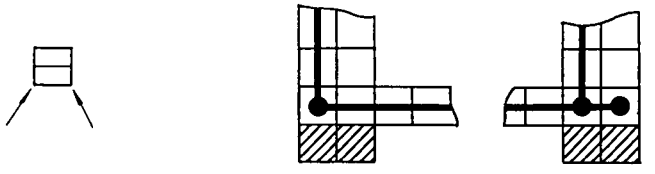
(b)　「左上端飾り」



(c)　「大うろこ」　「小うろこ」



(d)　「かどうろこ」



(e)　「左下端飾り」　「右下端飾り」

図 4.9　縦横直線の端点と線端飾りの関係

を示す。同図で，黒丸は直線の線端位置を示し，斜線を施した画素が線端飾りを構成する画素である。

4. 4. 1項では縦，横直線は，それぞれ構成する2本の黒ランとしてのみ抽出した。そこで線端飾りの制御を行なうために，2本の構成黒ランから縦，横直線の端点座標を一意的に決定することが必要となる。しかし，こゝで抽出すべき端点座標は，前述のように，隣接する2本の黒ランの始点，終点のいずれか，又は，両者が一致する端点についてのみ決定すれば良いため，あいまいさなく行なうことができる。

すなわち，式（4.13），（4.14），（4.15），（4.16）として抽出された縦，横直線に対して，次の縦，横直線が抽出される。

◦横直線；$\{(I_{k'}, J_{sk'}), (I_{k'}, J_{ek'})\}$ （4.24）

こゝで，$I_{k'}=I_h$，

かつ，

$$J_{sk'} = \begin{cases} J_{s1}\ ;\ (J_{s1}=J_{s2}\text{の場合}) \\ \text{不定}\ ;\ (J_{s1}\neq J_{s2}\text{の場合}) \end{cases}$$

$$J_{ek'} = \begin{cases} J_{e1}\ ;\ (J_{e1}=J_{e2}\text{の場合}) \\ \text{不定}\ ;\ (J_{e1}\neq J_{e2}\text{の場合}) \end{cases}$$

である。

◦縦直線；$\{(I_{sk}, J_k), (I_{ek}, J_k)\}$ （4.25）

こゝで，$J_k=J_v-1$

$$I_{sk} = \begin{cases} I_{s1}\ ;\ (I_{s1}=I_{s2}\text{の場合}) \\ \text{不定}\ ;\ (I_{s1}\neq I_{s2}\text{の場合}) \end{cases}$$

$$I_{ek} = \begin{cases} I_{e1}\ ;\ (I_{e1}=I_{e2}\text{の場合}) \\ \text{不定}\ ;\ (I_{e1}\neq I_{e2}\text{の場合}) \end{cases}$$

である。

今，1個の漢字パターンに対してh本の横直線とv本の縦直線が抽出されたとして，改めて，縦，横直線を次式で表わす。

◦横直線；$\{(I_{k'}, J_{sk'}), (I_{k'}, J_{ek'})\}$ （4.26）

$k'=1, 2, \cdots\cdots, h$

◦縦直線；$\{(I_{sk}, J_k), (I_{ek}, J_k)\}$ （4.27）

$k=1, 2, \cdots\cdots, v$

この時，線端飾りの種類の識別と付加位置は以下のようにして求めることができる。なお，式（4.26），（4.27）で$J_{sk'}$と$J_{ek'}$のいずれか1つと，I_{sk}とI_{ek}のいずれ

か１つは不定の場合があるが，前述のように，線端飾りの制御は確定座標値をもつ端点のみに対して行なわれる。

(1) 「あたま」

縦直線の始点（I_{sk}，J_k）（図 4.9 (a)の黒点）が，どの横直線とも接しないとき，すなわち，すべてのｋ′に対して

$$\left.\begin{aligned} &I_{sk} \neq I_{k'} \\ &\text{又は} \\ &J_k < J_{sk'} \text{ または } J_k > J_{ek'} \\ &(k' = 1, 2, 3, \cdots\cdots, h) \end{aligned}\right\} \quad (4.28)$$

のとき，「あたま」

$$a(I_{sk}+1, J_k+2) = 1 \quad (4.29)$$

を付ける。（図 4.9 (a)の斜線部）

なお，式（4.28）の条件だけでは，縦直線が斜線，曲線に接している場合にも「あたま」を付けることになるが，この場合，式（4.29）の「あたま」の殆んどは斜線，曲線に含まれてしまうことになる。

(2) 「左上端飾り」

縦直線の始点（I_{sk}，J_k）（図 4.9 (b)の黒点）に，いずれかの横直線の始点が一致するとき，すなわち，いずれかの k' に対して

$$\left.\begin{aligned} &I_{sk} = I_{k'} \\ &\text{かつ} \\ &J_k = J_{sk'} \end{aligned}\right\} \quad (4.30)$$

が成立するとき，左上端飾り

$$a(I_{sk}-1, J_k) \quad (4.31)$$

を付ける。（図 4.9 (b)斜線部）

(3) 「うろこ」

横直線の終点（$I_{k'}$，$J_{ek'}$）（図 4.9 (c)の黒点）が，どの縦直線とも接しないとき，すなわち，すべてのｋに対して，

$$\left.\begin{aligned} &J_{ek'} \neq J_k + 1 \\ &\text{又は} \\ &I_{k'} < I_{sk} \text{ または } I_{k'} > I_{ek} \end{aligned}\right\} \quad (4.32)$$

$$(k = 1, 2, \cdots\cdots, v)$$

のとき，「大うろこ」

$$\left.\begin{aligned} &a(I_k' - 1, J_{ek}' - 3) = 1 \\ &a(I_k' - 1, J_{ek}' - 2) = 1 \\ &a(I_k' - 1, J_{ek}' - 1) = 1 \\ &a(I_k' - 2, J_{ek}' - 2) = 1 \end{aligned}\right\} \qquad (4.33)$$

を付ける。

但し，「大うろこ」を付加することによって対象としている横直線が他の黒画素と連結される場合，すなわち

$$a(I_k' - 2, J_{ek}' - 3) \oplus a(I_k' - 2, J_{ek}' - 1) \oplus a(I_{ek}' - 1, J_{ek}' - 4) \oplus a(I_k' - 3, J_{ek}' - 2) = 1 \qquad (4.34)$$

（⊕は論理和を表わす）

ならば，「小うろこ」

$$a(I_k' - 1, J_{ek}' - 1) \qquad (4.35)$$

とする。（図 4.9 (c)斜線部）とする。

(4) 「かどうろこ」

横直線の終点（I_k', J_{ek}'）（図 4.9 (d)の黒点）にいずれかの縦直線の始点が一致するとき，すなわち，いずれかの k に対して

$$\left.\begin{aligned} &I_k' = I_{sk} \\ &\text{かつ} \\ &J_{ek}' = J_k + 1 \end{aligned}\right\} \qquad (4.36)$$

が成り立つとき，「かどうろこ」

$$\left.\begin{aligned} &a(I_k' - 1, J_{ek}' - 1) = 1 \\ &a(I_k' + 1, J_{ek}' + 1) = 1 \end{aligned}\right\} \qquad (4.37)$$

を付ける。（図 4.9 (d)斜線部）

(5) 「左下端飾り」，「右下端飾り」

縦直線の終点（I_{ek}, J_k）（図 4.9 (e)の黒点）が，いずれかの横直線の始点か終点に一致するとき，すなわち，いずれかの k' に対して

$$\left.\begin{aligned} & I_{ek} = I_k' \\ & \text{かつ} \\ & J_k = J_{sk}' \end{aligned}\right\} \qquad (4.38)$$

か，又は

$$\left.\begin{aligned} & I_{ek} = I_k' \\ & \text{かつ} \\ & J_k = J_{ek}' - 1 \end{aligned}\right\} \qquad (4.39)$$

が成立するとき，「下端飾り」

$$a(I_{ek}+1,\ J_k) = 1 \qquad (4.40)$$

と，

$$a(I_{ek}+1,\ J_k+1) = 1 \qquad (4.41)$$

を付ける。（図 4.9 (e)斜線部）

4.4.4 変換処理結果

以上，横直線を細線化し，線端飾りを付加することによって，ゴシック体から得られた明朝体の例を図 4.8 (b)に示す。

他の字種に対する変換結果の例を図 4.10 に示す。

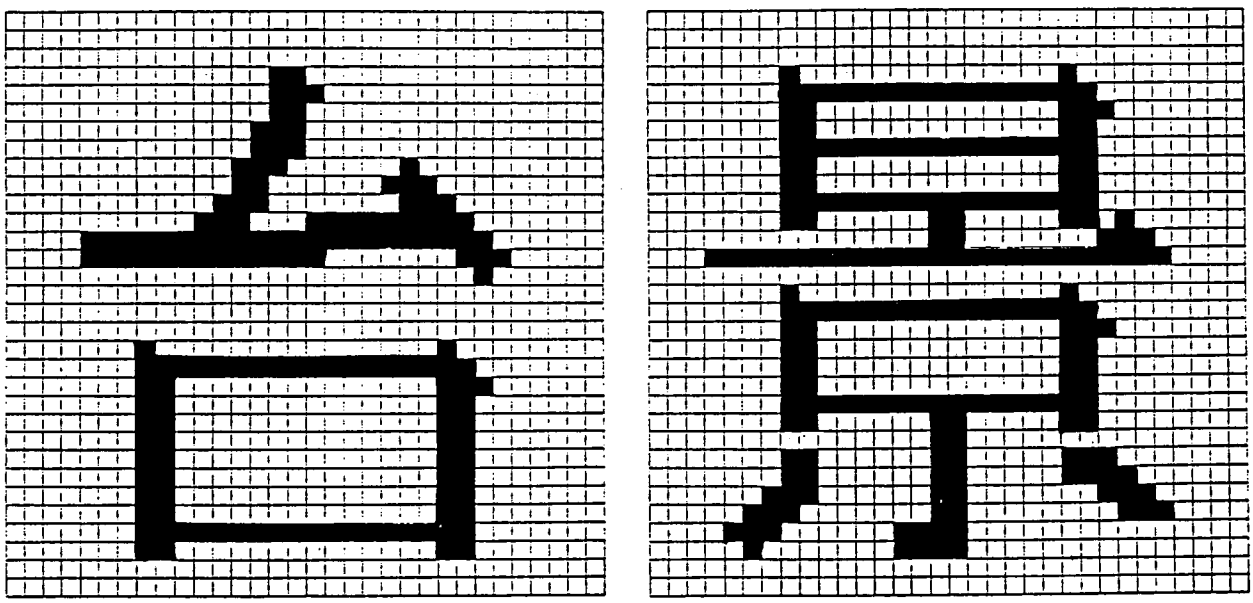


図 4.10　変換明朝体の例（その 1）

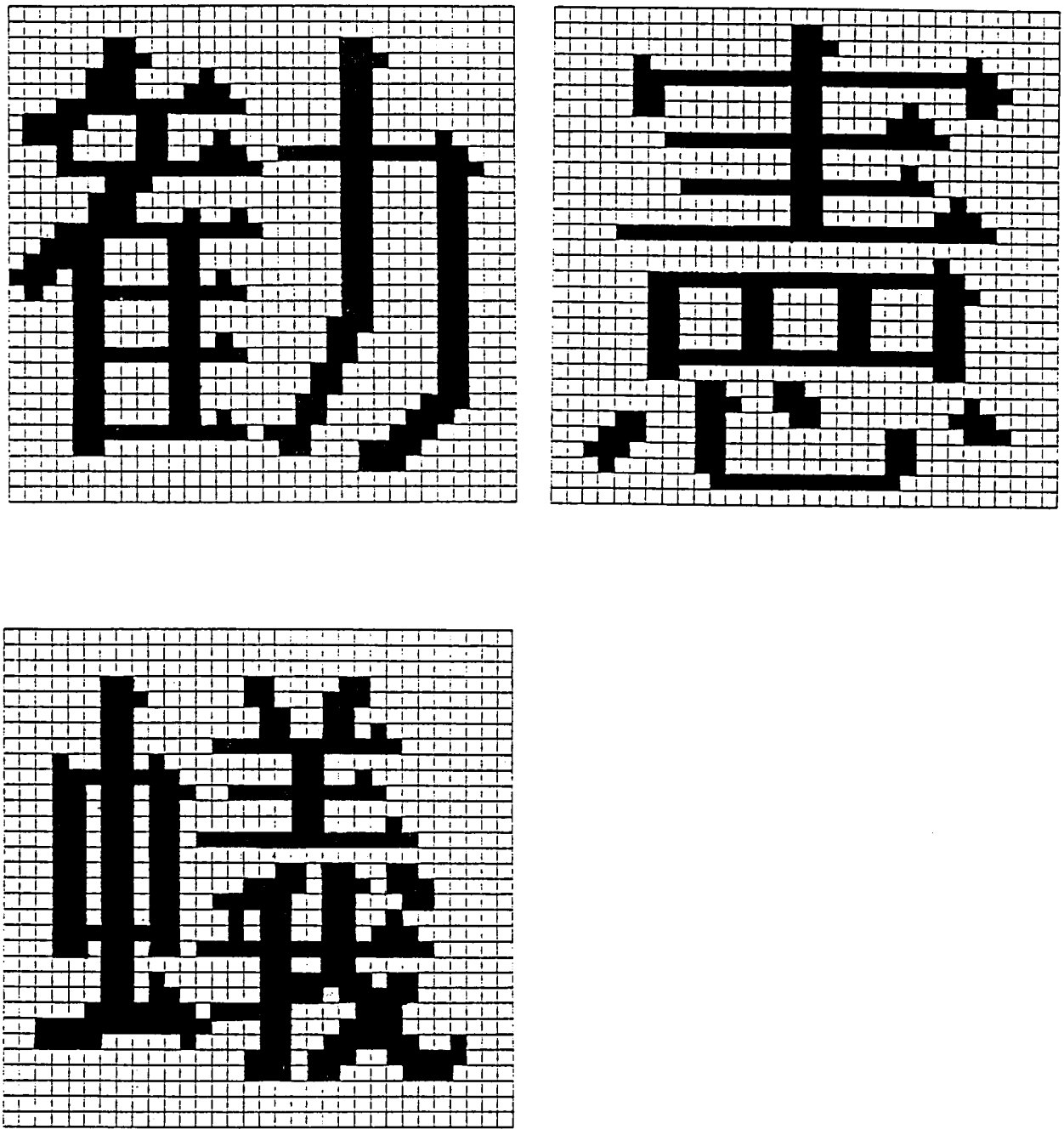


図 4.10　変換明朝体の例（その２）

4.5 書体変換処理の評価

以上述べてきた，明朝体からゴシック体への変換と，その逆の変換処理について，書体変換の達成度と処理量について述べる。

4.5.1 書体変換の達成度

人手によって個々に作成された明朝体とゴシック体の漢字パターンを原漢字パターンとして，4.3節と4.4節で述べた明朝体とゴシック体との書体間の変換処理によって得られた変換明朝体と変換ゴシック体の漢字パターンを比較することによって書体変換処理の効果を評価する。

表4.2に明朝体とゴシック体漢字パターンについて，それぞれ原漢字パターンと変換漢字パターンの間の差分パターンの黒画素数（以下，差分画素数とよぶ。）を評価法1として示す。同表で絶対差分画素数とは原漢字パターンでの両書体間の差分黒画素数を意味する。

表4.2　差分画素数

試料漢字	絶対差分画素数	差分画素数			
		評価法1		評価法2	
		ゴシック体	明朝体	ゴシック体	明朝体
台	55	27	60	27	28
価	50	35	41	7	10
景	109	57	50	27	19
勧	86	38	55	26	37
憲	116	17	33	15	20
蟻	78	51	49	43	36
平均	82.3	37.5	48.0	24.2	25.0

この表から，漢字「台」のゴシック体から明朝体への変換処理の場合を除いて，差分画素数は絶対差分画素数よりも少なくなっていることが分かる。

書体変換処理の評価指数として次の変換達成率を定義する。

$$\text{変換達成率} = \frac{\text{絶対差分画素数} - \text{差分画素数}}{\text{絶対差分画素数}} \times 100 \qquad (4.42)$$

変換達成率が高い程，原漢字パターンに近い変換漢字パターンが得られたことになる。試料の6字に対しては，明朝体からゴシック体への変換では55％，ゴシック体から明朝体への変換では42％の変換達成率となる。

図4.11に個々の試料についての変換達成率を図で示している。

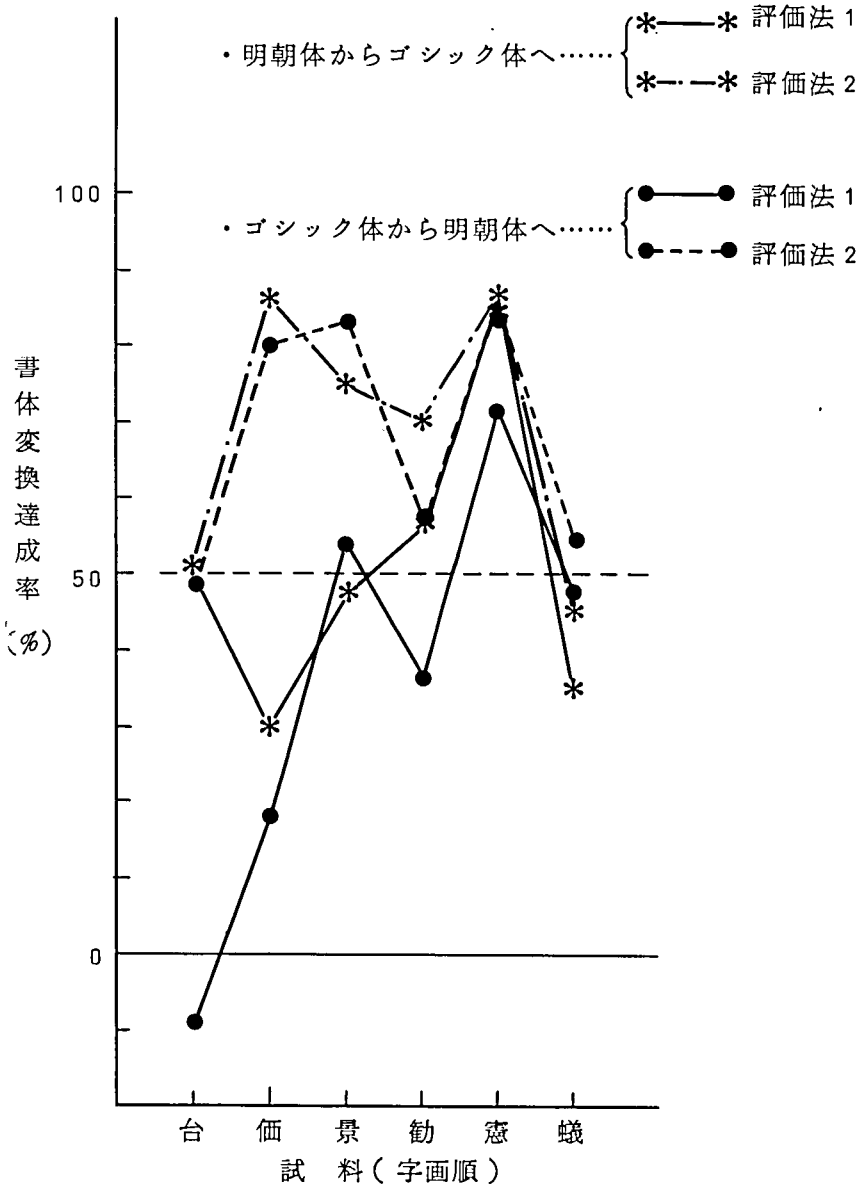
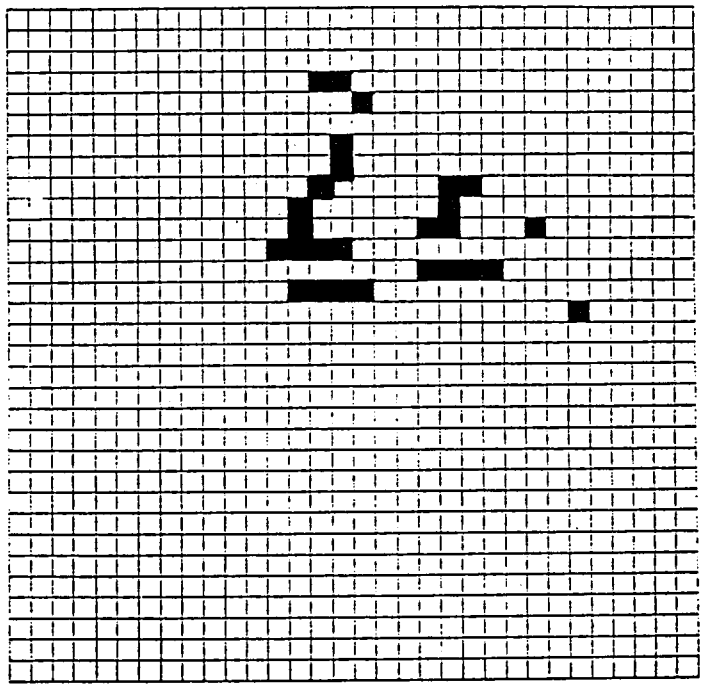
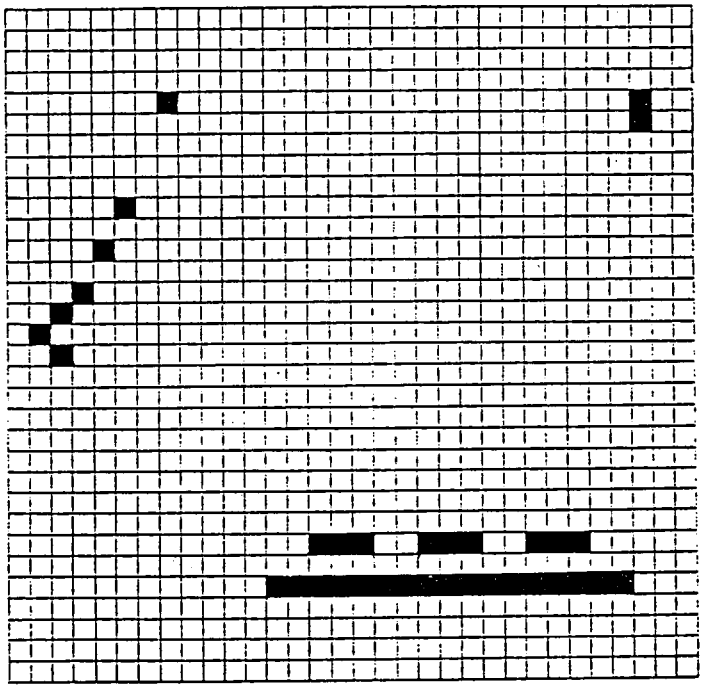


図 4.11　書体変換の達成率

また，図4.12，図4.13にそれぞれゴシック体と明朝体について原漢字パターンと変

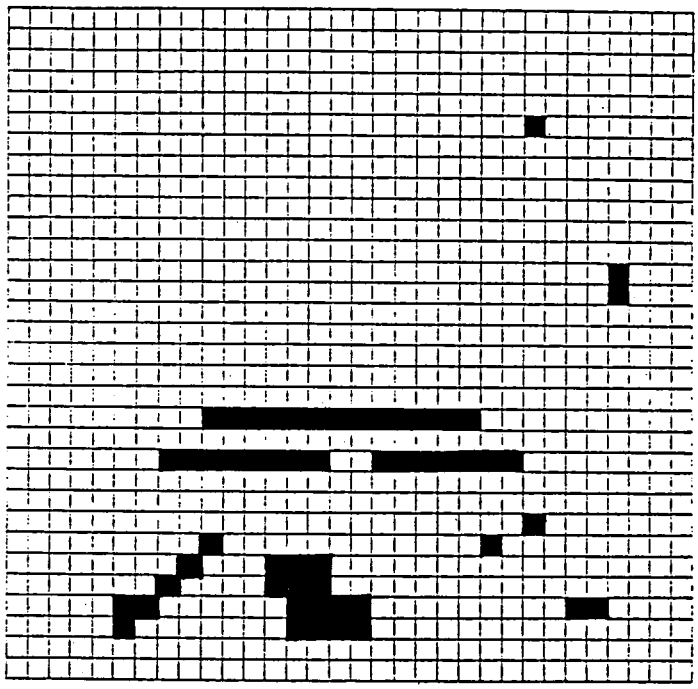


(a)　「台」の差分パターン

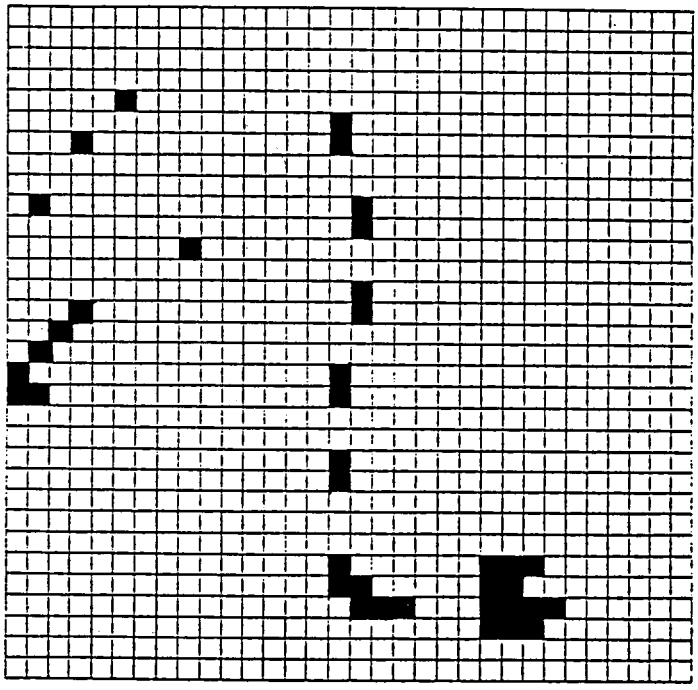


(b)　「価」の差分パターン

図 4. 12　ゴシック体に対する原パターンと変換パターンの差(1)

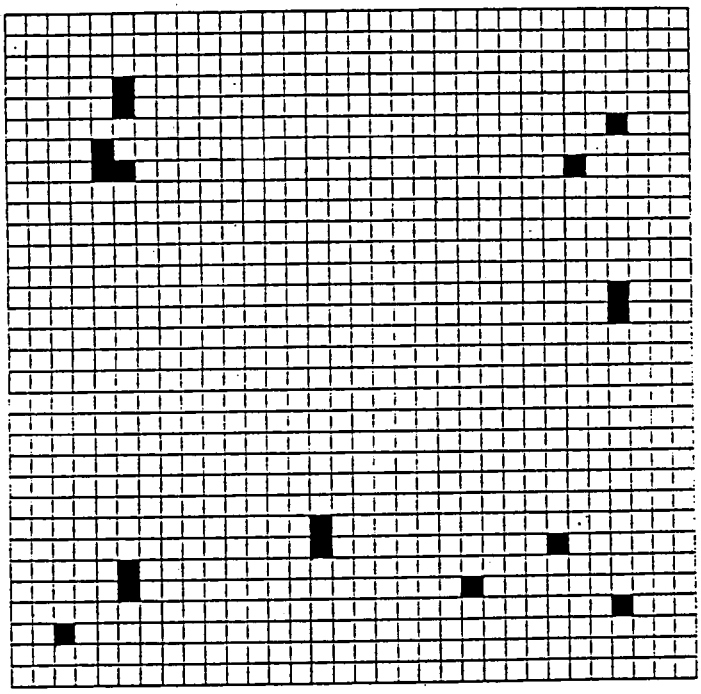


(c)　「景」の差分パターン

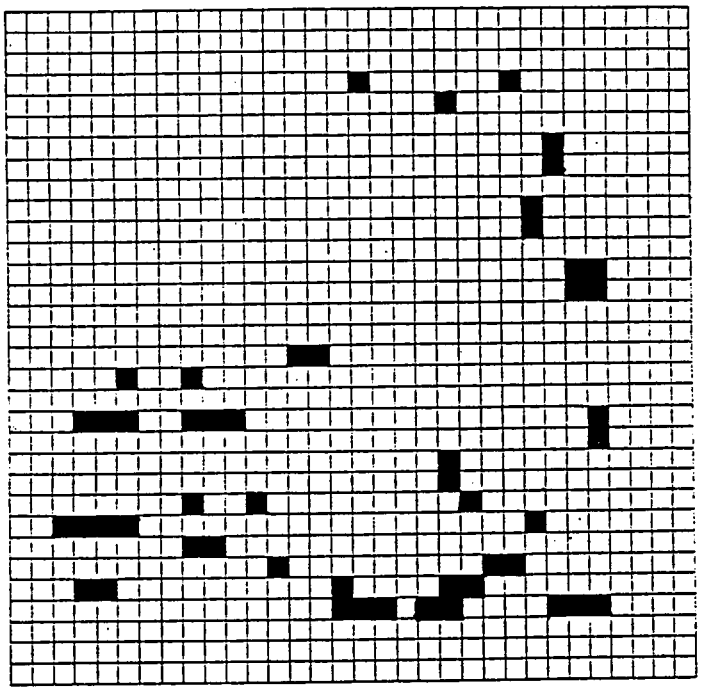


(d)　「勧」の差分パターン

図 4. 12　ゴシック体に対する原パターンと変換パターンの差(2)

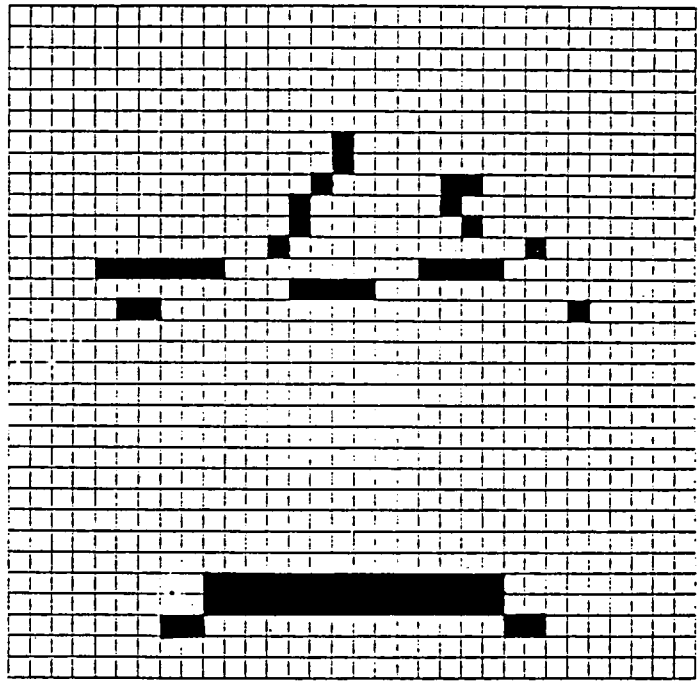


(e)　「憲」の差分パターン

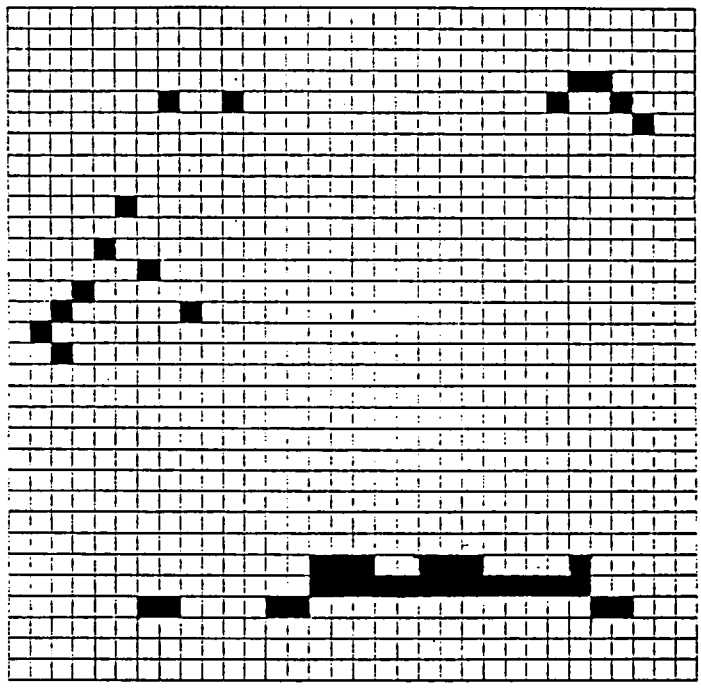


(f)　「蟻」の差分パターン

図 4. 12　ゴシック体に対する原パターンと変換パターンの差(3)

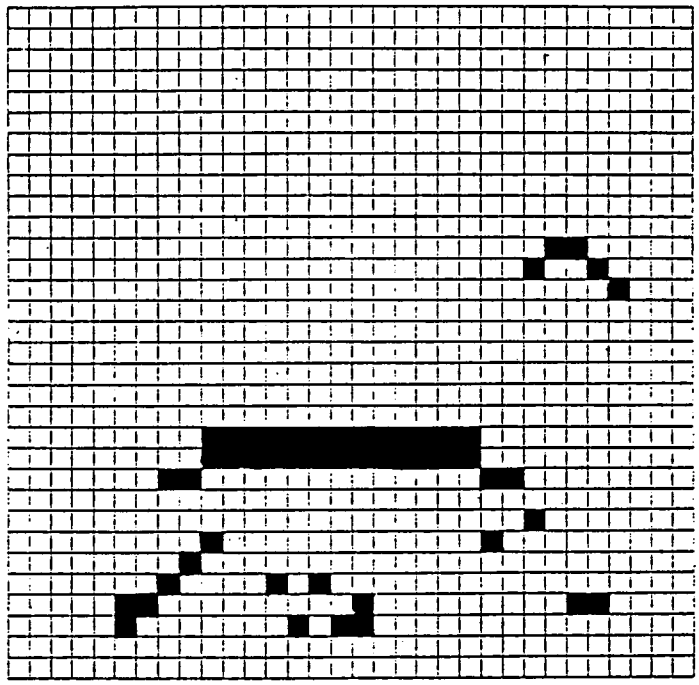


(a)　「台」の差分パターン

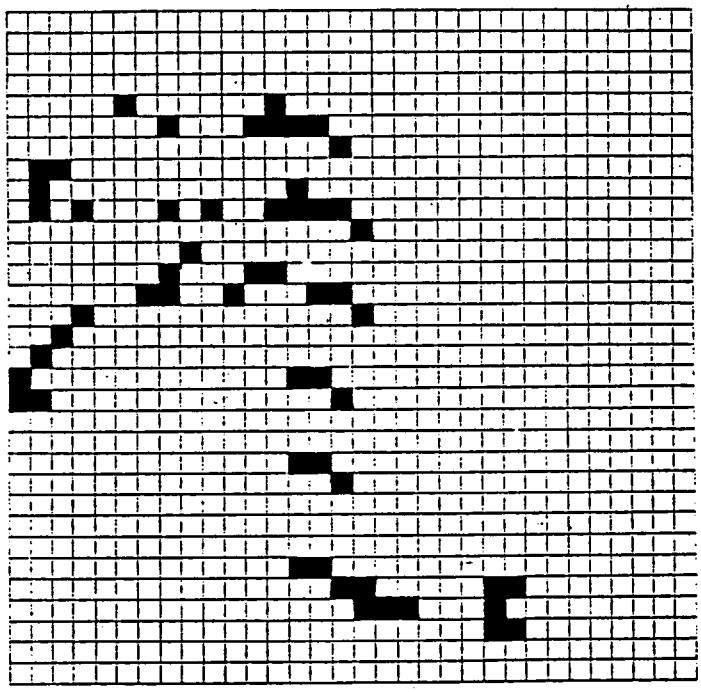


(b)　「価」の差分パターン

図 4. 13　明朝体に対する原パターンと変換パターンの差(1)

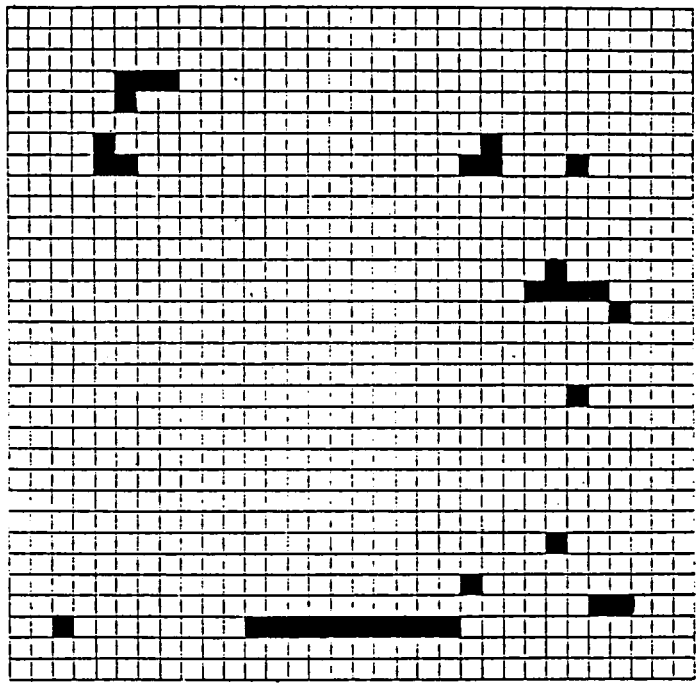


(c)　「景」の差分パターン

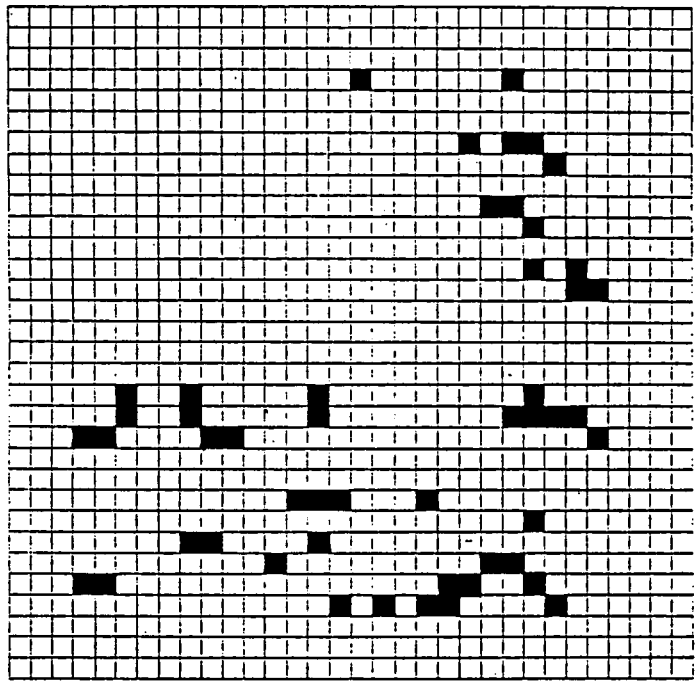


(d)　「勧」の差分パターン

図 4. 13　明朝体に対する原パターンと変換パターンの差(2)



(e)　「憲」の差分パターン



(f)　「蟻」の差分パターン

図 4. 13　明朝体に対する原パターンと変換パターンの差(3)

換漢字パターンの差分パターンを示している。この差分パターンから，変換達成率を下げている主要な要因として，以下の3つがあげられる。

(i) 斜線，曲線部に対する未対処

(ii) 両書体間での横直線の位置（縦方向）および長さのずれ

この横直線の位置および長さのずれは次のような大幅な差分画素数の増加をもたらす。

① 明朝体からゴシック体への変換

位置ずれに対しては横直線の長さの2倍，長さのずれに対しては，ずれの2倍の差分画素の増加

② ゴシック体から明朝体への変換

位置ずれに対しては横直線の長さの2倍，長さのずれに対しては，ずれ分に加えて，線端飾りの位置ずれ分の差分画素の増加

(iii) 線端飾りの1つである「はね」に対する未対処

以上の3つの要因の中で横直線の位置，長さのずれが最も大きく変換達成率を低下させている。

そこで

① 横直線の位置および長さのずれは1画素に過ぎず視覚的に大きな差異を生じない。

② 本試料の漢字パターンは素人の作成によるものであり，位置および長さをずらす必然性が疑問である。

③ 横直線の位置ずれは横直線の分布密度が小さく，位置選択の自由度が大きな部分で生じており，予じめ書体変換処理に適したデザインを行なう上で不都合は生じない。

の3点から，横直線の位置および長さの1画素分のずれは両書体間の本質的な差異でないと考えられる。

そこで，横直線の位置および長さの1画素分のずれに基づく差分画素を無視して書体変換処理を評価する。これを評価法2として，差分画素数および変換達成率を表4.2と図4.11に示している。

この結果，明朝体からゴシック体，およびその逆変換に対して，試料6字に対する平均で，それぞれ71％と70％の変換達成率が得られる。

また，個々の漢字パターンで見れば，「憲」，「景」，「価」のような縦横直線の割合が大きい漢字パターン程，変換達成率は高く，逆に，斜線，曲線の割合が大きな「蛾」，「勧」の変換達成率は低くなる。特に，「台」に含まれる緩やかな斜線部では差分画素数が多く，変換達成率を下げることが分かる。

書体変換処理は，その処理結果の変換漢字パターンをそのまま出力パターンとして用いることも出来るし，データ圧縮のための前処理としての適用も可能である。

例えば，明朝体漢字パターンを基にゴシック体漢字パターンを発生させる場合を考えると，書体変換処理を前処理として適用すれば，平均24.2個の位置のみを変換制御情報として記憶すれば良い。この結果，差分画素の位置座標（Ｉ，Ｊ）を2進符号で表現したとしても，（32×32）漢字パターンとしてそのまま記憶する場合に比較すると約24％のメモリ量で済むことになる。

4.5.2 処理量と変換速度

図4.3に示したフローに従った書体変換処理の処理量の概略値を（32×32）の漢字パターンについてフォートランのダイナミックステップ数で表4.3に示す。

表4.3 書体変換処理量

変換	処理内容	ダイナミックステップ数
明朝体からゴシック体への変換	縦横直線の分離	15.0 k
	横線幅の拡大	3.0 k
	線端飾りの除去	4.5 k
	再合成	2.0 k
	計	24.5 k
ゴシック体から明朝体への変換	縦横直線と端点座標の検出	16.0 k
	横直線の細線化	2.0 k
	線端飾りの付加	0.5 k
	計	18.5 k

明朝体からゴシック体への変換処理が逆の変換処理に比較して，線端飾り除去のための縦横直線の分離，飾り除去ウィンドウ処理，再合成の過程でダイナミックステップ数が増加している。

1フォートランステップを5アセンブラステップに換算し，1 μ秒／アセンブラステップの小形計算機を想定すると，書体変換速度は明朝体からゴシック体への変換が約40字／秒，ゴシック体から明朝体への変換が約50字／秒となる。

処理量の多くは縦横直線の抽出のためのランレングスの算出に要しているため，この単純処理をハードウェア化することが高速化に対しては不可欠となる。

4.6 まとめ

従来，書体の異なる漢字パターンを発生するためには発生すべき全ての書体の漢字パターンをパターンメモリ内に格納しておくことが必要であり，漢字パターン発生器は極めて高価なものとなった。

また，この問題を解決するための画素形漢字パターンの書体変換処理に関する研究例は本研究以前には見当らなかった。

本章では漢字パターン発生器の経済化の技術の1つとして漢字書体の自動変換処理の可能性について考察した。

漢字書体の変換は字形を構成するストロークの形状の変化を伴なうものであり，基本的には画素形漢字パターンからのストロークの抽出が必要である。しかし，画素形漢字パターンからストロークを抽出するためには複雑な処理が必要であり，それでも，なお，完全なストロークの抽出は現状では困難である。

本研究では，サイズ（32×32）程度以下の漢字パターンにおける明朝体とゴシック体のの書体間の主要な差異である縦，横直線の線幅比と線端飾りの有無の制御が，簡易な縦，横直線の抽出処理で可能であることを示した。

すなわち

(1) 明朝体からゴシック体への変換では，

① 線幅の変更が必要な横直線のみを，横方向の黒ランレングスの閾値処理により抽出することで線幅比の制御が可能であること。

② 明朝体からゴシック体への変換では線端飾りの種類と位置の識別は不要であるため，連結する黒画素の個数の閾値処理として線端飾りの除去が可能であること。

(2) ゴシック体から明朝体への変換では

③　線幅の変換が必要な横直線の抽出は前記①と同じく黒ランレングスの閾値処理で可能であること。

④　線端飾りを付加するためには，線端飾りの種類と付加位置を決定しなければならないが，それは，縦，横直線の端点の状態を識別することにより可能であること。

⑤　線端飾りの制御に必要な縦，横直線の端点は，直線を形成する2本の黒ランの始点又は終点が一致する端点のみで良く，両端が決まった完全な直線を抽出する必要がないこと。

を示した。

また，書体毎に個々に人手で作成された漢字パターンを原漢字パターンとして，原漢字パターンと変換漢字パターンとの差の黒画素数を用いて書体変換達成率を定義し，書体変換の達成度を評価した。

この結果，縦，横直線幅の比，斜線・曲線の形状の差および線端飾りの有無を明朝体とゴシック体の書体間の基本的な差異とする時，両書体間の差分画素数に換算して約70％の書体変換の達成率が得られることを示した。

今後，さらに，書体変換の達成率を向上させるためには，斜線・曲線部の形状変換処理が不可欠であり，そのためには画素形漢字パターンからのストロークの完全な抽出とストロークの表現形式の標準化が残された課題となる。

付 録 4.1　書体変換処理の検討で用いた試料の字種

円，五，物，勢，球，毛，階，価，桑，憲，縄，城，界，鮫，喉，制，堀，台，羽，宰
臓，煙，蟻，派，髪，於，景，請，覚，剰，裟，勧，内，零，卯，幽，仮，椿，敏，繁

第5章　漢字パターンの作成

5.1　ま　え　が　き

漢字パターンの出力，表示に先立って，それらの漢字パターンを作成することが必要である。

従来，漢字パターンはデザイナーらの人手によって作成されており，数千字種から成る漢字パターンのセットを作り上げるには多大の時間と労力，それに経費を必要とした。

漢字パターンの種類が少ない場合には，人手による作成で間に合わせることが可能であったが，最近のようにＣＲＴ受像機，ファクシミリ受信機，インクジェットプリンタなど解像度の異なる多種類の漢字出力装置が現われ，それぞれに適した漢字パターンを作成し，提供するためには人手による作成では対応できなくなってきた。

以上のような情況を背景として，計算機と人間がディプレイ画面を介して会話処理を行なうことにより，効率的に漢字パターンを作成する研究が報告されている。(45)(46)

それらで検討された漢字パターン作成装置では，漢字母形を必要なサイズの画素形漢字パターンに変換して人間に提示し，人間からの修正指示を受けて，漢字パターンの画素状態を修正するという基本的な会話処理機能を実現している。

本研究では，計算機との会話処理を行なうという基本的な考え方は従来の研究に学び，さらに，人間の作成労力を極力削減するための自動整形処理について考察する。すなわち，漢字母形をディジタル化した際に生じる線分周辺のノッチ除去と線幅の規格化という２つの基本的な整形処理について検討する。

また，この自動整形処理を導入するに際して，その機能をより効率的に働かせるために従来の装置構成法に比較して，人間と機械の機能分担の考え方をさらに進めた。

すなわち，漢字母形を２値化する際の閾値処理や自動整形処理で，処理結果の漢字パターンの品質を左右するパラメータについては，同一パターンに対してパラメータ値を変えて複数通りの処理結果を装置側で求め，人間はその複数個の処理結果について優劣の判定のみを行なえば良い構成とした。

また，漢字パターンの処理としては，前章までは処理の対象とした漢字パターンが，既に整形処理を終えて，ノッチのない，線幅が統一されたパターンであったのに対して，本

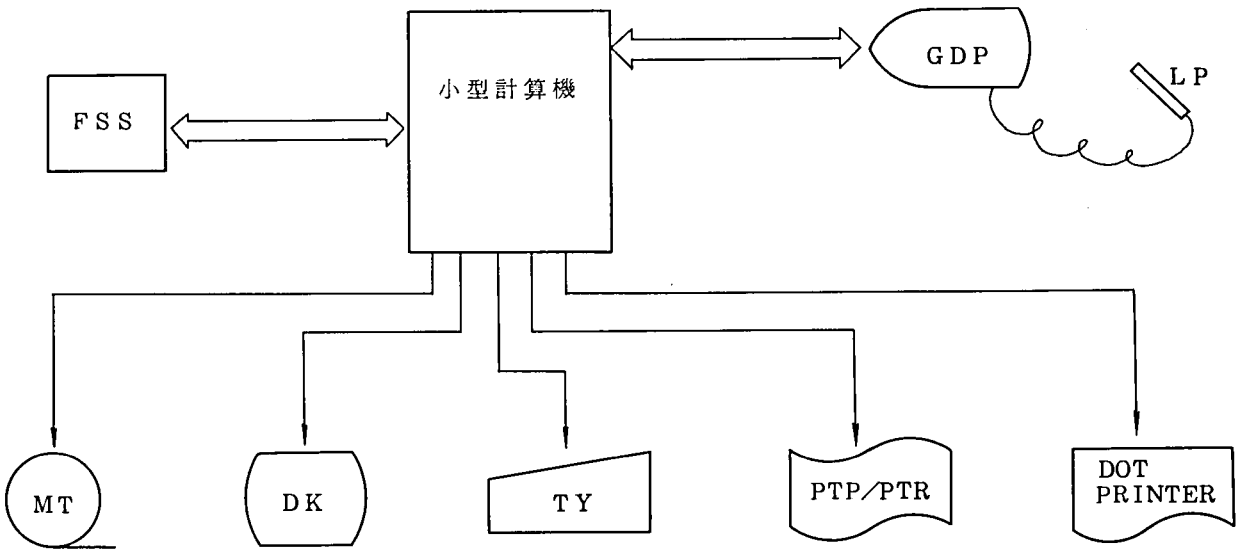
章で扱かう漢字パターンは，ノッチをもった，線幅の不揃いな未整形処理パターンである点が基本的に異なる。さらに，縦横直線に関しては，両端が確定した完全な直線として抽出する必要がある点も大きく異なる。

5.2 漢字パターン作成装置の構成と機能概要

ここでは漢字パターン作成装置の構成と機能の概要について述べ，人間と機械の機能の分担法と，次節以降で述べる漢字パターンの整形処理の装置内における位置づけを明らかにする。

(1) 装置構成

図 5.1 に本研究で対象とした漢字パターン作成装置の構成図を示す。



FSS：フライング・スポット・スキャナ　　DK　：磁気ディスク装置
GDP：グラフィック・ディスプレイ　　LP　：ライトペン
TY　：タイプライタ　　PTP/PTR：紙テープリーダ・パンチャ
MT　：磁気テープ装置

図 5.1　漢字パターン作成装置の構成

本装置は小形計算機を中心にグラフィックディスプレイ装置（以下，ＧＤＰと略記する），フライングスポットスキャナ（以下，ＦＳＳと略す），補助記憶装置（磁気ディスク装置），タイプライタ，ドットプリンタ，紙テープパンチャから成っている。

各構成装置の主要な処理内容と仕様を付録 5.1 に示す。

(2) 主要機能と作成処理の流れ

本漢字パターン作成装置はカード上に印刷された漢字を母形パターンとし，ＧＤＰ装置を介して計算機と人間とが会話処理を行ない，所要のサイズをもつ漢字パターンを作成する機能をもつものである。

特に本装置の構成に際しては，「人間の作業労力をできるだけ軽減すること」を基本方針としている。

この基本方針を実現するために，本装置では，従来報告されている装置では見られなかった，次の２つの機能を新たに導入した。

① ノッチを除去し，線幅を統一する自動整形処理機能

② 処理過程における処理パラメータ値の人間による選択機能

②は各処理過程で，同一漢字パターンに対して，処理パラメータ値を変えて複数回の処理を計算機で実行し，人間は，その複数個の処理結果の中から最も品質の良いものを選択，指示するものである。この機能により，カード上の漢字母形の印刷濃度むらやＦＳＳのシェーディングに基づく，２値化過程での最適閾値の変動や，後述する整形処理過程での字種の差による最適パラメータ値の変化に対処することが可能となり，自動整形処理機能を有効に作用させるための，装置構成方式上の裏付け手段となる。

同一処理を複数回繰り返すため，計算機による処理時間が増大し，会話処理における応答時間の増大は避けられないが，オペレータによる修正時間の短縮と作業内容の質的な改善が図られる。

図 5.2 に漢字パターンの作成処理の流れを示している。

すなわち，

① カード上に印刷された漢字母形をＦＳＳを用いて（９６×９６）の２値画素パターンとして読み取る。（以後，この漢字パターンを読み取り漢字パターンと略す。）文字カードの例を図 5.3 に示す。

② （９６×９６）の読み取り漢字パターンを（３×３）のサブマトリクスに分割し，各サブマトリクス内の黒画素の個数に関して閾値処理を行ない，（３２×３２）の標準サイズの漢字パターンを作る。なお，この時，閾値を４通り変えて４個の標準サイズの漢字パターンを作成する。

以後，この処理を前処理と呼ぶ。

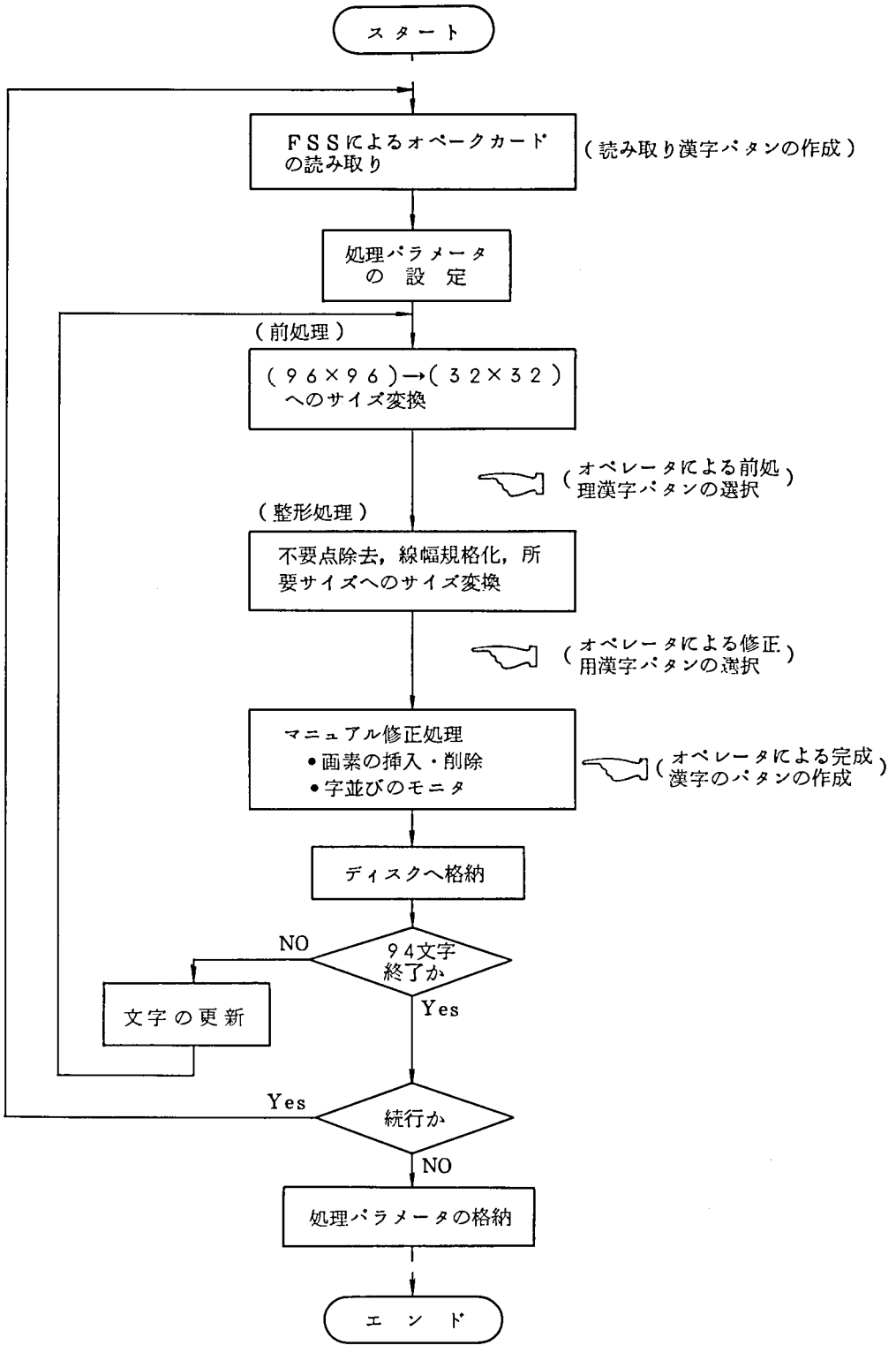
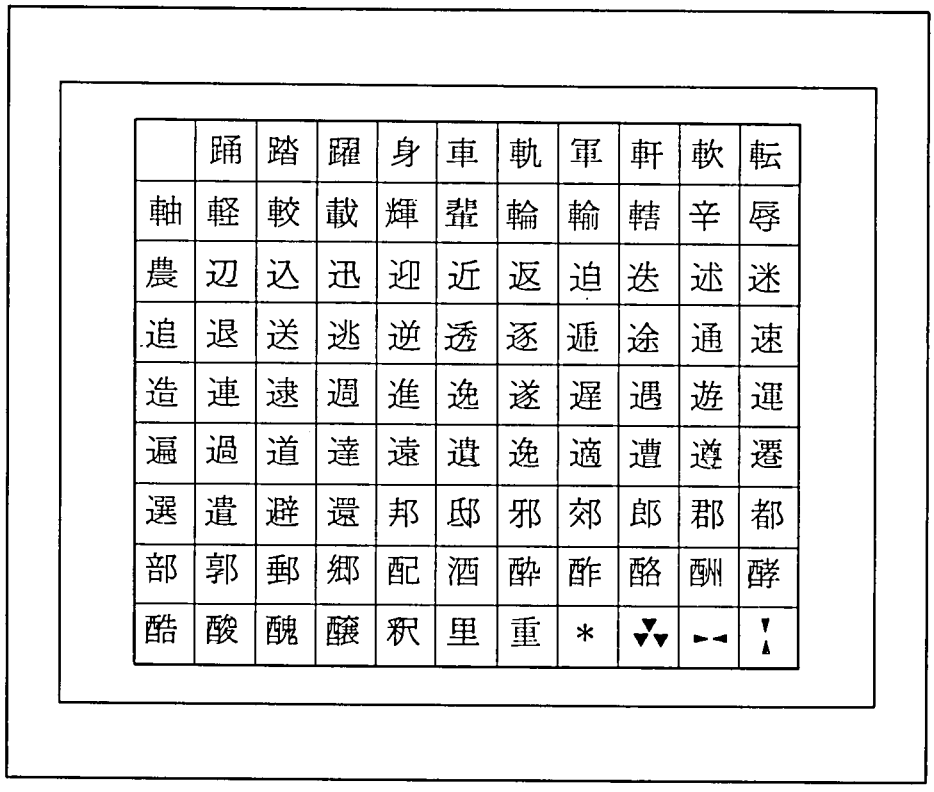


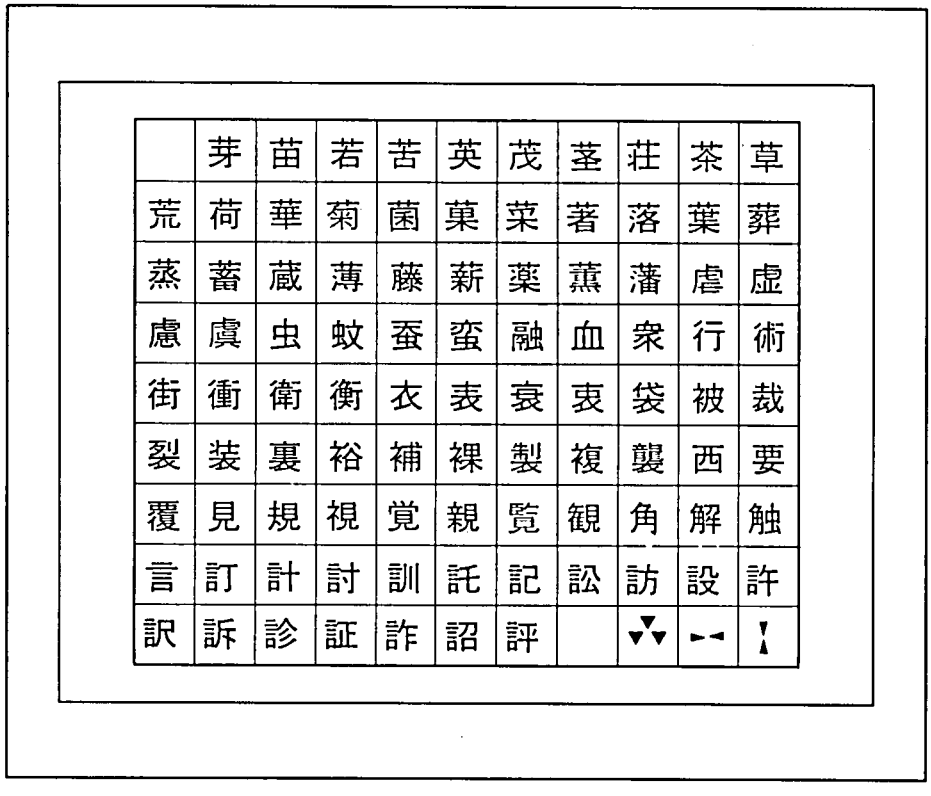
図 5.2　漢字パターン作成処理の流れ

③　4個の処理結果の漢字パターンをＧＤＰ装置上に表示し，最も品質が良い漢字パターンを人間が選択し指示する。（以後，選択された漢字パターンを前処理漢字パターンと



	踊	踏	躍	身	車	軌	軍	軒	軟	転
軸	軽	較	載	輝	輩	輪	輸	轄	辛	辱
農	辺	込	迅	迎	近	返	迫	迭	述	迷
追	退	送	逃	逆	透	逐	逓	途	通	速
造	連	逮	週	進	逸	遂	遅	遇	遊	運
遍	過	道	達	遠	遺	逸	適	遭	遵	遷
選	遣	避	還	邦	邸	邪	郊	郎	郡	都
部	郭	郵	郷	配	酒	酔	酢	酪	酬	酵
酷	酸	醜	醸	釈	里	重	*	▼▼	►◄	!

(a) 明 朝 体



	芽	苗	若	苦	英	茂	茎	荘	茶	草
荒	荷	華	菊	菌	菓	菜	著	落	葉	葬
蒸	蓄	蔵	薄	藤	薪	薬	薫	藩	虐	虚
慮	虞	虫	蚊	蚕	蛮	融	血	衆	行	術
街	衝	衛	衡	衣	表	衰	衷	袋	被	裁
裂	装	裏	裕	補	裸	製	複	襲	西	要
覆	見	規	視	覚	親	覧	観	角	解	触
言	訂	計	討	訓	託	記	訟	訪	設	許
訳	訴	診	証	詐	詔	評		▼▼	►◄	!

(b) ゴ シ ッ ク 体

図 5. 3 文字カードの例

呼ぶ。）

④　前処理漢字パターンに対して，処理パラメータ値を3通りに変えて，ノッチ除去と線幅の統一に関する整形処理を行なう。同時に，必要に応じて，サイズ変換を実行する。以後，この処理を整形処理と呼ぶ。

⑤　3通りの整形処理結果と整形処理を行なわない前処理漢字パターンの4つの漢字パターンをＧＤＰ装置上に表示し，最も品質の良い漢字パターンをオペレータが選択する。（以後，選択された漢字パターンを修正用漢字パターンと呼ぶ。）

⑥　修正用漢字パターンをＧＤＰ装置上に表示し，ライトペンを用いて，画素の追加，削除およびパターン位置の修正を行なう。（以後，修正処理を終えた漢字パターンを完成漢字パターンと呼ぶ。）

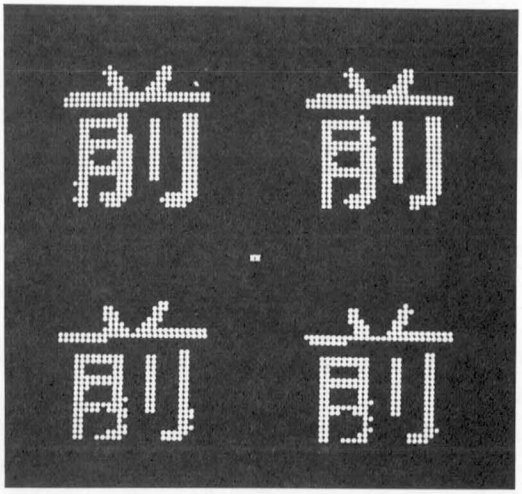
⑦　完成漢字パターンを磁気ディスク装置へ格納する。

⑧　②へもどり，次の漢字パターンの作成を行なう。

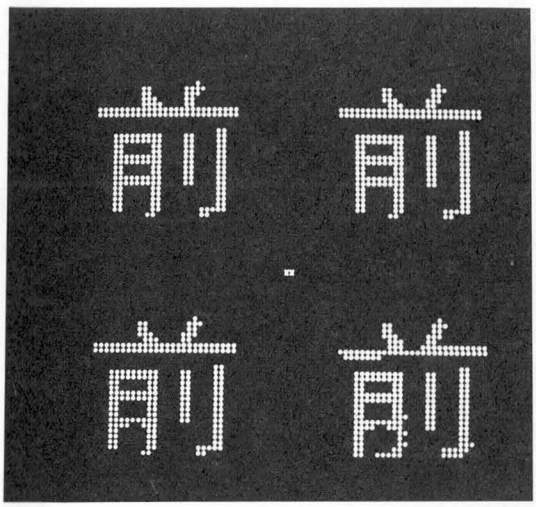
以上の処理を繰り返し，必要な漢字パターンの作成が終了し，磁気ディスク装置へ格納した後，

⑨　漢字コードを入力し，目的の漢字パターンを検索し，ドットプリンタや紙テープへ漢字パターンデータを出力する。図5.4に漢字パターン作成時の一連の表示画面例を示す。

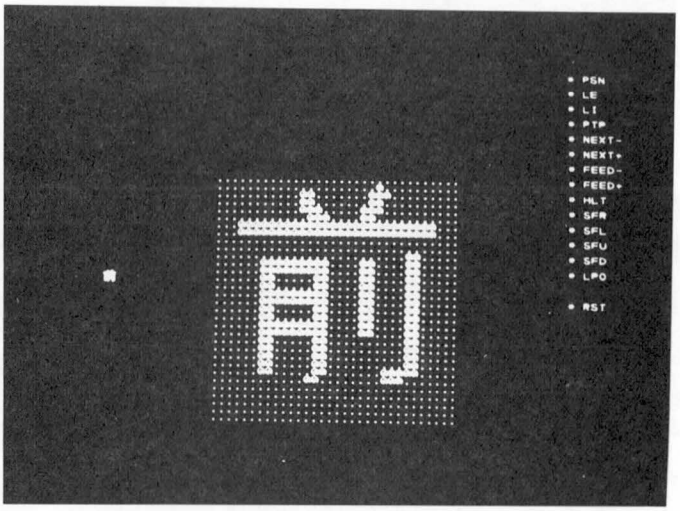
(a)　ライトペンを用いた会話処理



(b)　4個の標準サイズ漢字
パターンの表示画面
（前処理結果の表示）



(c)　3個の整形処理結果と未整形
処理（右下）の漢字パターン
の表示画面



(d)　画素単位の修正用画面

図 5. 4　表示画面の例

5.3 整 形 処 理[79][80][81]

5.3.1 前 処 理

カード上の漢字母形はＦＳＳを用いてサイズ（９６×９６）の読み取り漢字パターンとして読み取られた後，前処理により（３２×３２）の前処理漢字パターンへと変換される。

すなわち，サイズ（９６×９６）の読み取り漢字パターンを

$$A(I, J)=\begin{cases}0：白画素\\1：黒画素\end{cases} \qquad (5.1)$$

$$1 \leqq I, J \leqq 96$$

とする時，次式により（３×３）のサブマトリクス内の黒画素の個数（０～９）を濃度値とする（３２×３２）

$$a'(i, j)=\sum_{\substack{I=1,2,3\\J=1,2,3}} A\{30(i-1)+I, 30(j-1)+J\} \qquad (5.2)$$

$$(1 \leqq i, j \leqq 32)$$

の多値パターンへと変換し，この多値パターンを閾値TH（$0 \leqq TH \leqq 9$）を用いて２値化することにより，次の前処理漢字パターンを得る。

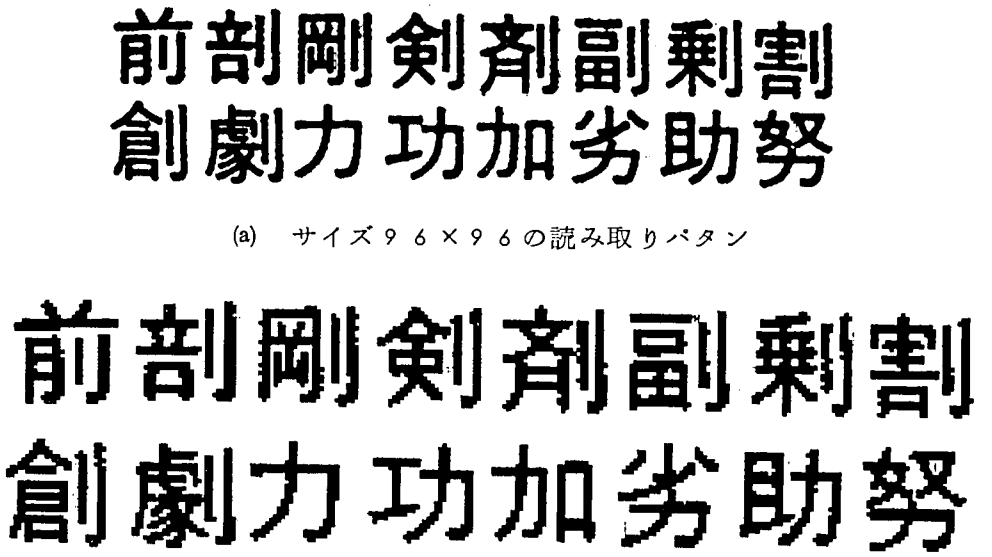
$$a(i, j)=\begin{cases}0 : for\ a'(i, j)<TH\\1 ; for\ a'(i, j)\geqq TH\end{cases} \qquad (5.3)$$

$$(1 \leqq i, j \leqq 32)$$

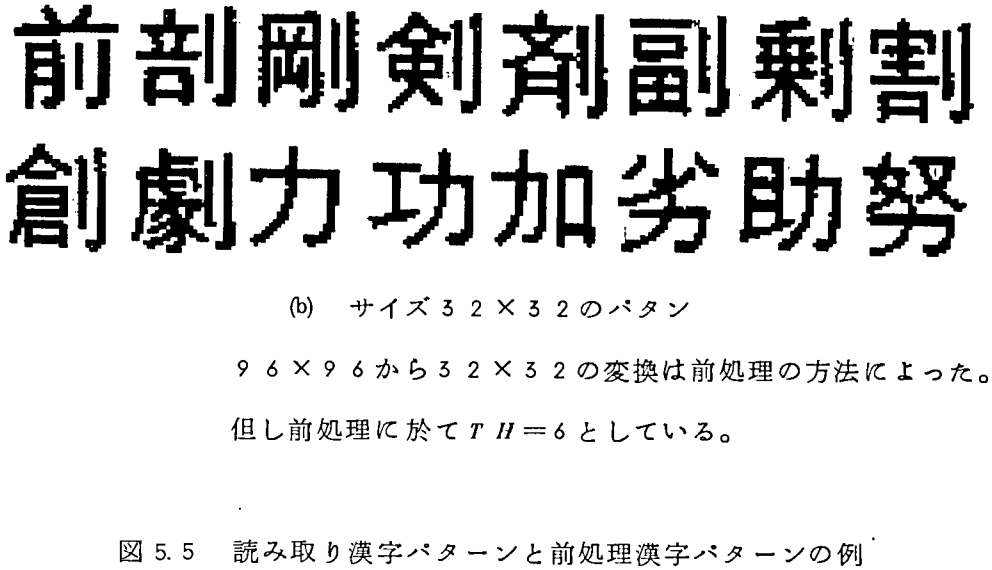
この前処理において，最適閾値THは漢字母形の印刷濃度むらや，ＦＳＳの状態によって変動し，一意的に設定しておくことは出来ないため，本装置では既述のように，４通りの閾値に対する処理結果の中から，人間が最適値を選択する方法を用いた。

図5.5の(a)，(b)に，それぞれ読み取り漢字パターンと前処理漢字パターンの例を示す。この図から分かるように，オペレータの判断により，前処理における最適閾値を設定したとしても，線分周辺のノッチと線幅の不揃いを完全に除去することは出来ない。

また，このノッチの除去と線幅の修正は人間による会話修正作業の大きな割合いを



(a)　サイズ９６×９６の読み取りパタン



(b)　サイズ３２×３２のパタン

９６×９６から３２×３２の変換は前処理の方法によった。

但し前処理に於て$TH=6$としている。

図 5.5　読み取り漢字パターンと前処理漢字パターンの例

占めることになる。

そこで，ノッチの除去と線幅を統一する整形処理は，漢字パターンの作成の効率化に重要となる。

5.3.2　整形処理の考え方

整形処理の対象とす゛前処理漢字パターンを以後，原漢字パターンとよぶ。

原漢字パターンに含まれるノッチや線幅の不揃いは，原漢字パターンから漢字を構成するストローク情報が完全に抽出できれば除去することが出来る。なぜなら，抽出されたストローク情報を用いて新たに完全な形の漢字パターンを再発生させればよい。また，この時には，漢字パターンのサイズ変換や書体変換も容易に実現できることは第３章，第４章で示した。

整形処理で対象とする原漢字パターンは，未だ整形処理を終えてないことが，３章，４章で対象とした漢字パターンとは基本的に異なる。

すなわち，整形処理における原漢字パターンはノッチ，線幅の不揃いを含んでいる

ため全てのストローク情報を正しく抽出することは困難である。

そこで，次に述べる理由により縦横直線だけを抽出しストロークとして処理し，残りの斜線，曲線は画素パターンのままで処理する，縦横直線と斜線曲線の分離処理を整形処理の基本方針とした。

(1) 縦横直線を抽出する理由

① 縦横直線は漢字パターンの主要部を占める。従って，縦横直線に付随するノッチと線幅の不揃いは文字品質を大きく劣化させる。

② サイズ（３２×３２）程度の漢字パターンでは縦横直線に近い直線は全て完全な縦横直線として表現されており抽出が容易である。

③ 縦横直線の端点座標を用いることによりサイズ変換や書体変換の処理が可能である。

(2) 斜線，曲線をストロークとして抽出しない理由

① 画素配列の乱れは縦横直線部で目立ち，斜線，曲線部ではあまり目立たない。

② 斜線，曲線の表現は多様であり，一意的に形状を定めることが出来ない。

図 5.6 に漢字パターンの整形処理の流れを示す。大きく次の４つの処理から成っている。

① 縦横直線と斜線，曲線の分離

この縦横直線と斜線，曲線の分離精度が漢字パターンの整形処理全体の効果を左右する。分離処理の詳細は次節で述べる。

② 縦横直線部処理

分離された縦横直線から始点と終点を求め，それらの始点，終点を用いて所定の幅の縦横直線を新たに発生することにより，線幅が統一された縦横直線部を作成する。

③ 斜線，曲線部処理

原漢字パターンから縦横直線部を分離した後には斜線，曲線と縦横直線に付随していたノッチが残されるので，これらの内ノッチだけを除去する。

④ 再 合 成

線幅が統一された縦横直線部とノッチが除去された斜線，曲線部を合成し整形処理パターンを得る。

また，必要に応じて，②，③でサイズ，書体変換処理を行なう。

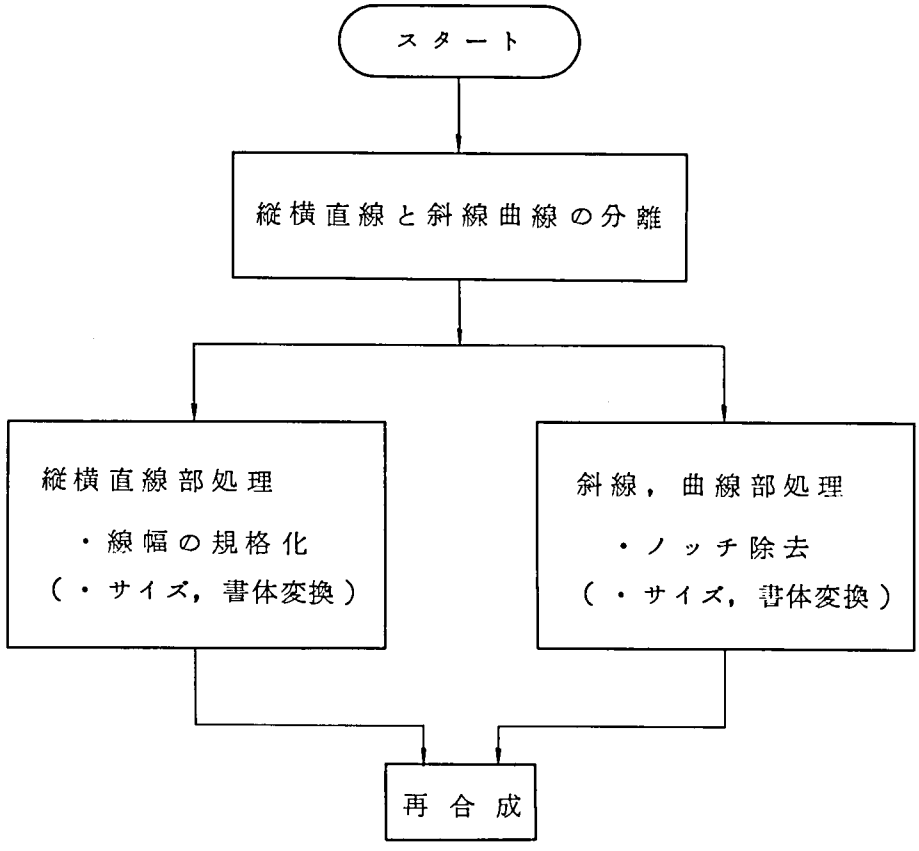


図 5.6　整形処理の流れ

5.3.3　縦横直線と斜線，曲線の分離

図 5.7 に縦横直線と斜線，曲線の分離処理の流れを示している。

縦横直線は，先ず，ランと呼ぶ黒画素の連なりを候補として抽出した後，そのランに対して識別処理を行なうことにより抽出する。

(1)　ランの抽出

縦方向または横方向の黒画素の連なりをランとし，その黒画素数をラン長とする。図 5.8 にランの例を示す。この図は 3 本の縦ラン a，b，c と 2 本の横ラン d，e で構成され，それぞれのラン長は１０，９，１２，１１，１３である。

原漢字パターンからラン長がある閾値 R_{th} 以上のランの始点座標（ i_{sk}, j_{sk} ）と終点座標（ i_{ek}, j_{ek} ）を抽出する。このときこのランを

$$\{ (i_{sk}, j_{sk}), (i_{ek}, j_{ek}) \} \qquad (5.4)$$

と表わすことにする。

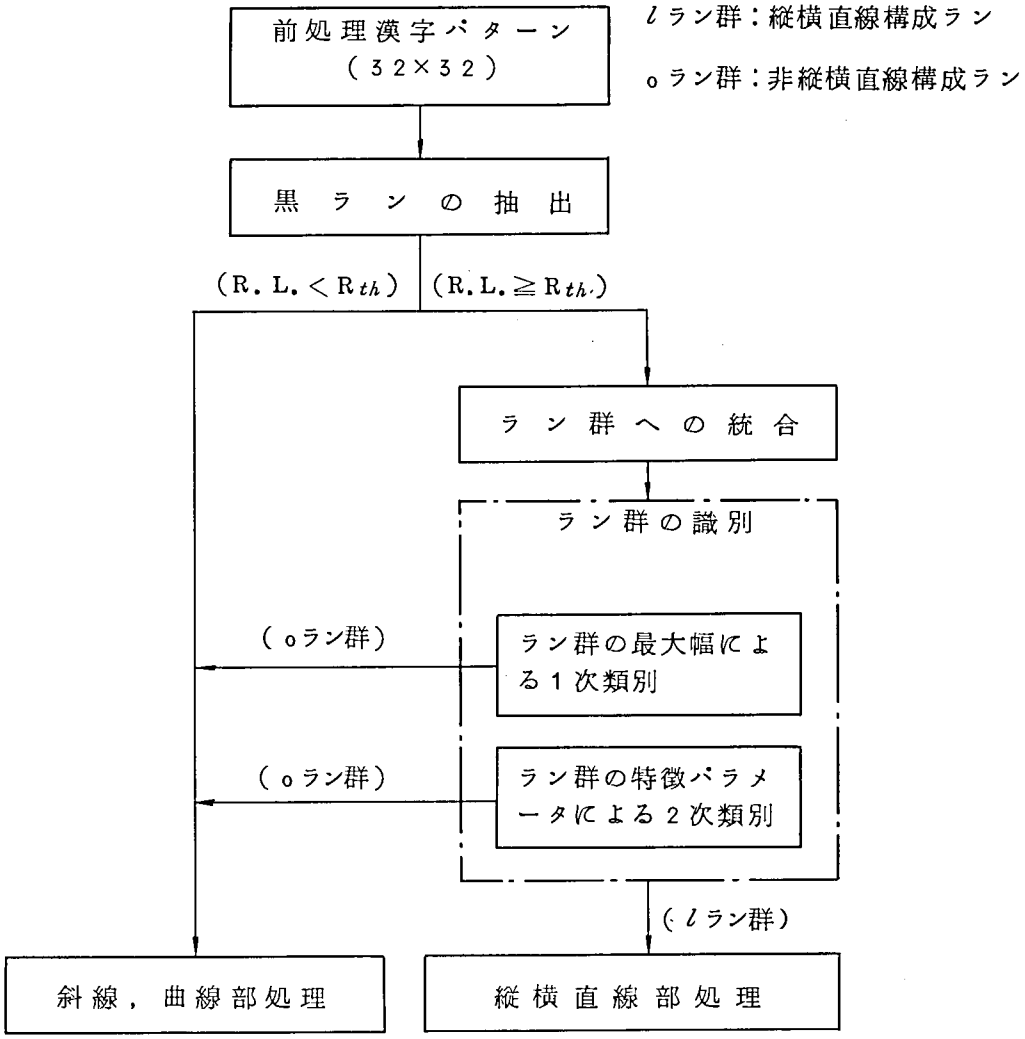


図 5.7　縦横直線と斜線曲線の分離処理の流れ

なお，座標系は図 5.8 に示す通りである。図 5.9 (b)は同図(a)を原漢字パターンとして，閾値 $R_{th}=7$ で抽出したランの例である。

なお，閾値 R_{th} の最適値は字種によって異なるため，漢字パターン作成装置では人間による最適値選択対象パラメータとしている。

(2)　ランのラン群への統合

互いに隣接するランの集合を 1 個のラン群として統合し，ラン群の最大幅を q とする。

r 本の縦ラン

$$\{(i_{s,k}, j_{s,k}), (i_{e,k}, j_{e,k})\} \qquad (5.5)$$

$k=1, 2, \cdots\cdots, r$

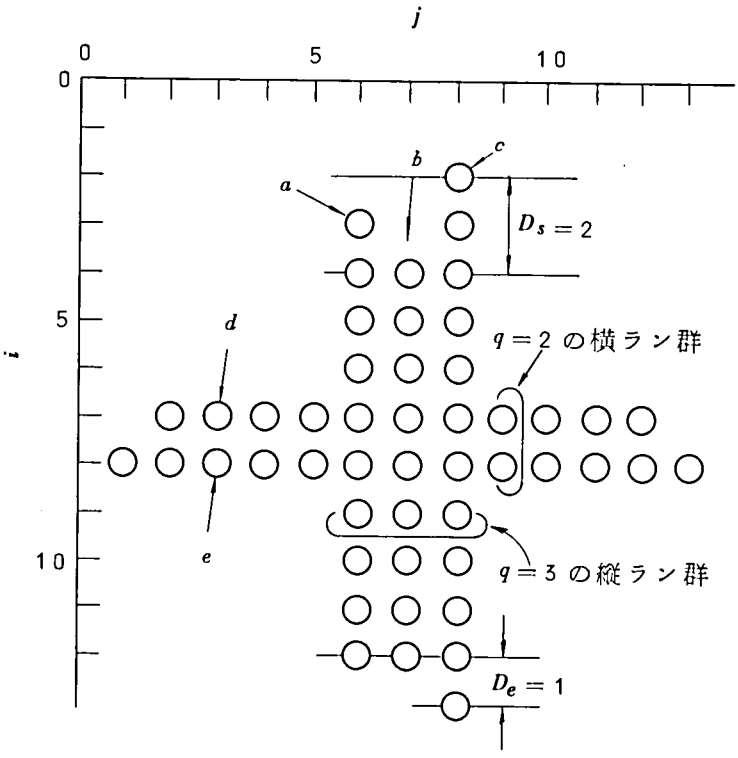
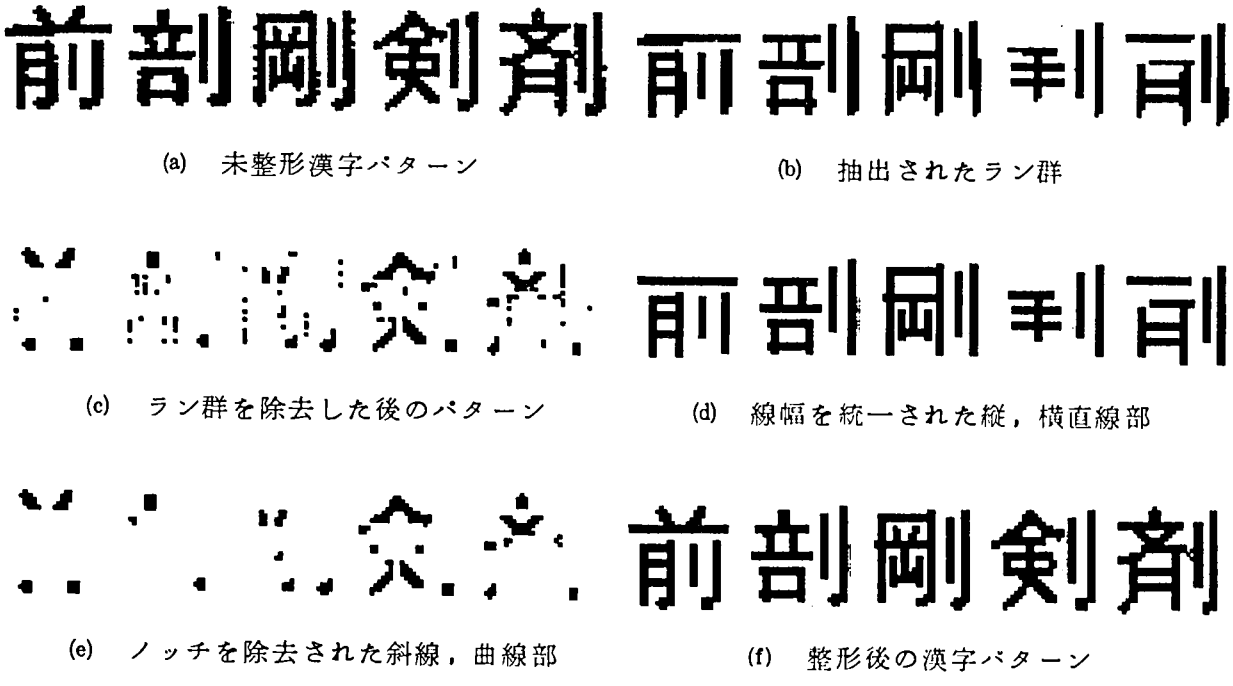


図 5.8　ラ　ン　の　例



(a)　未整形漢字パターン　　(b)　抽出されたラン群

(c)　ラン群を除去した後のパターン　　(d)　線幅を統一された縦，横直線部

(e)　ノッチを除去された斜線，曲線部　　(f)　整形後の漢字パターン

図 5.9　各整形処理過程におけるパターンの例

についてラン群を構成する条件は以下のとおりである。

〔ラン群の構成条件〕

式（5.5）で与えられるランの中の任意のラン

$$\{ (i_{s,k'}, j_{s,k'}), (i_{e,k'}, j_{e,k'}) \} \qquad (5.6)$$

に対して，次の2つの条件を同時に満たすラン

$$\{ (i_{s,k''}, j_{s,k''}), (i_{e,k''}, j_{e,k''}) \} \qquad (5.7)$$

が，式（5.5）で与えられるランの中に必ず存在すること。

① 横方向の座標値が連続していること。

$$j_{k'} = j_{k''} + 1 \quad 又は \quad j_{k'} = j_{k''} - 1 \qquad (5.8)$$

② ランの長さ方向に重なりを持つこと

$$\left.\begin{array}{l} i_{sk''} \leqq i_{s,k'} \leqq i_{e,k''} \\ 又は \\ i_{sk''} \leqq i_{e,k'} \leqq i_{e,k''} \end{array}\right| \qquad (5.9)$$

以上の構成条件を満足するラン群に対して，ラン群の最大幅qを次で定義する。

$$q = (j_k の最大値) - (j_k の最小値) + 1 \qquad (5.10)$$

以上，縦ラン群について述べたが横ラン群についても同様である。

図5.8の例では縦ラン群と横ラン群の最大幅はそれぞれ3と2である。

(3) ラン群の識別

適切な閾値R_{th}を用いて抽出されたラン群は縦横直線を構成する殆んどのランと，斜線，曲線を構成する一部のランを含むことになる。

この時，斜線，曲線の一部を縦横直線と見なして整形処理を行なえば原漢字パターンにない損傷を処理結果に加え，逆に，人間による修正作業量を増やすことになる。

そこでラン群を縦横直線から抽出されたラン群か斜線，曲線から抽出されたラン群かを識別することが重要となる。以下，このラン群の識別法について考察する。なお，縦横直線から抽出されたラン群をlラン群，斜線，曲線から抽出されたラン群をoラ

ン群として，それぞれサフィックス l ，o を付して表わす。

(i) 最大幅 q による 1 次類別

原漢字パターン内の縦，横直線の幅は一定の範囲内にある。従って，l ラン群の最大幅 q は一定の範囲内におさまり，その範囲外の最大幅値をとるラン群は必ず o ラン群になる。この性質を利用してラン群を o ラン群に属するものと，o ラン群または l ラン群に属するものに類別できる。この類別を 1 次類別と呼ぶ。

付録 5.2 に示すサイズ（３２×３２）の明朝体漢字パターン９４文字を選びサンプルパターンとし，サンプルパターンから，閾値 $R_{th}=7$ に対して抽出された全ラン群を l ラン群と o ラン群に人手により類別した結果について，ラン群の最大幅と各群に含まれるラン群の個数分布を図 5.10 に示す。

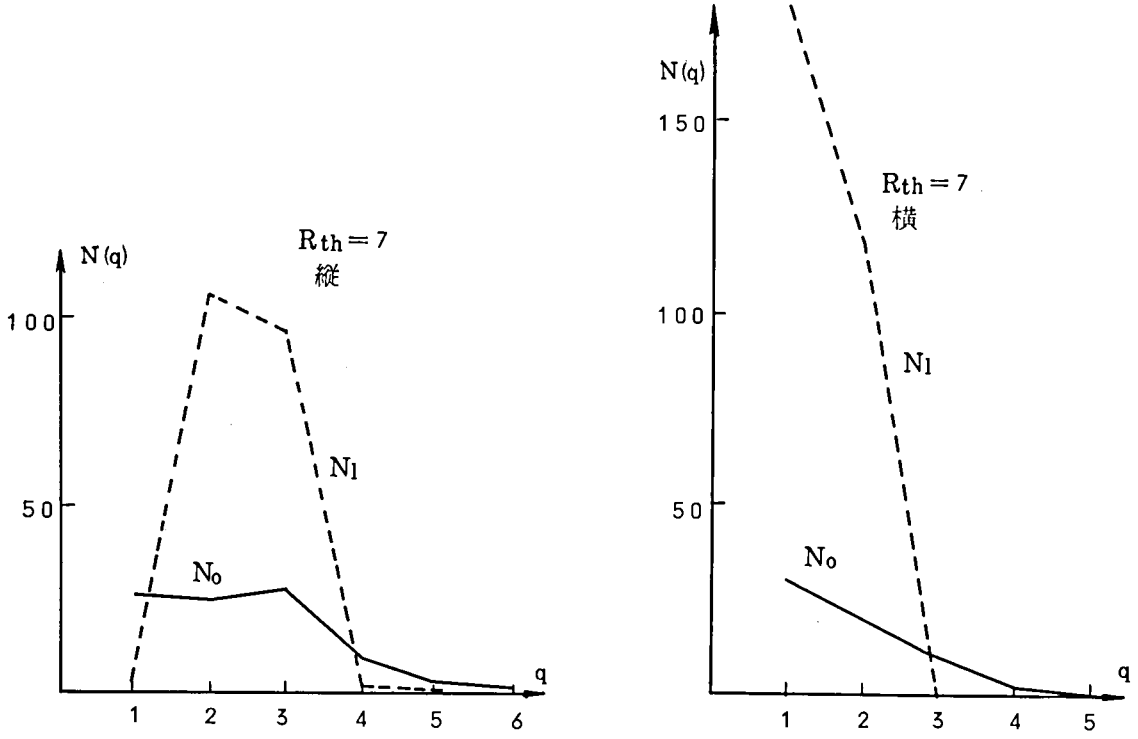


図 5.10 ラン群の最大幅 q と出現頻度 N

図で，$N_l(q)$，$N_o(q)$ はそれぞれ最大値幅 q の時の l ラン群と o ラン群の個数である。

この結果からサンプルパターンでは，縦ラン群では $q=1$ または $q \geqq 4$ のとき，横ラン群では $q \geqq 3$ のとき，それぞれ o ラン群であると識別できる。これ以外の場

合には，最大幅 q だけでは o ラン群か l ラン群かを確定することは出来ない。

そこで，q＝2，3 の縦ラン群と q＝1，2 の横ラン群に対しては，次の 2 次類別を行なう。

(ii) ラン群の特徴パラメータ

パラメータの選定においては，なるべく簡単な処理で求まり，且つ，ラン群の種類によりその値が有意な差を示すものでなければならない。

ラン群の特徴を示すパラメータとしてはランの傾き，始点最大差，終点最大差，それにラン群の平均長が考えられる。ラン群の平均長はラン群抽出時の閾値 R_{th} として考慮されるので，他の 3 つのパラメータについて，前記のサンプルパターンを用いて，2 次類別パラメータとしての有効性を検討する。

① ラン群の傾き A

縦ラン群を代表する直線と横ラン群を代表する直線を最小二乗法で求め，それぞれ式（5.11）と（5.12）とおく。

$$j = A_v \cdot i + B_v \qquad (5.11)$$

$$i = A_h \cdot j + B_h \qquad (5.12)$$

ここで，A_v, A_h はそれぞれ縦ラン群の傾きと横ラン群の傾き，B_v, B_h はそれぞれラン群の切片である。座標系は図 5.8 に示した通りである。

ラン群を構成する各画素と式（5.11）と（5.12）の直線との距離の 2 乗和を最小とするように傾き A_v, A_h を定める。当然，完全な縦，横直線のラン群に対しては，$A_v = A_h = 0$ となる。

ラン群の傾きの導出にはラン群の構成画素の全てについてその座標値を用いる必要はなく，ラン群中のランの始点と終点の座標値だけを用いればよい。

式（5.5）で表わされる縦ラン群について，その傾きを求めると次のようになる。

$$A_v = \frac{\sum_{k=1}^{r} X_k \cdot \sum_{k=1}^{r} (j_k \cdot Y_k) - \sum_{k=1}^{r} Y_k \cdot \sum_{k=1}^{r} (j_k \cdot X_k)}{\sum_{k=1}^{r} X_k \cdot \sum_{k=1}^{r} \{Z(i_{e,k}) - Z(i_{s,k}-1)\} - (\sum_{k=1}^{r} Y_k)^2} \qquad (5.13)$$

ここで

$$X_k = (i_{e,k} - i_{s,k} + 1) \qquad (5.14)$$

$$Y_k = X_k \cdot (i_{s,k} + i_{e,k})/2 \qquad (5.15)$$

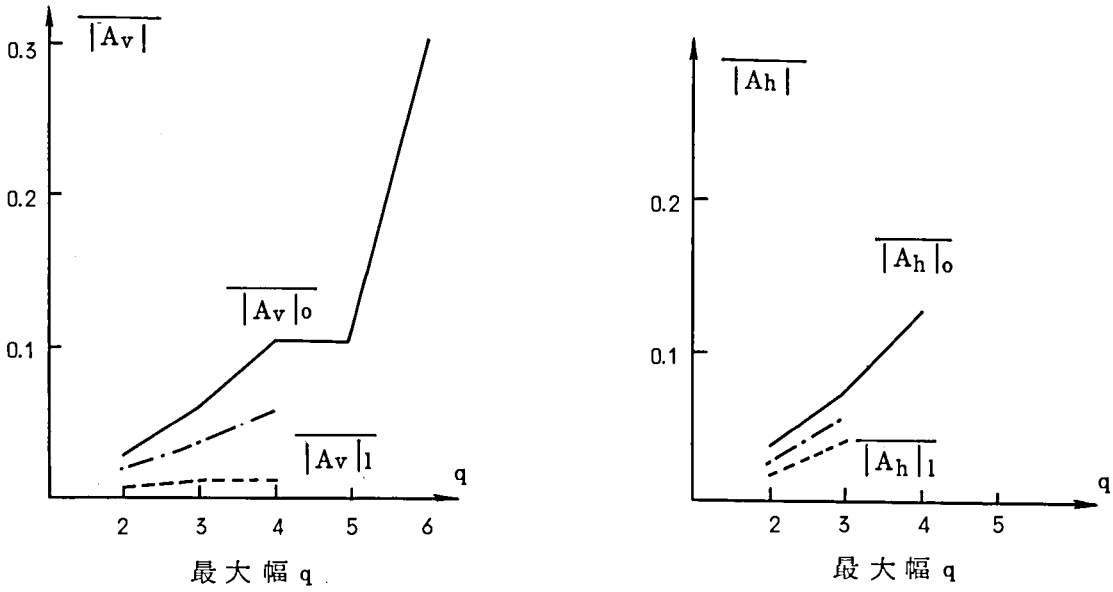
$$Z(i) = i \cdot (i+1) \cdot (2i+1)/6 \qquad (5.16)$$

である。

導出の詳細については付録 5.3 に示す。

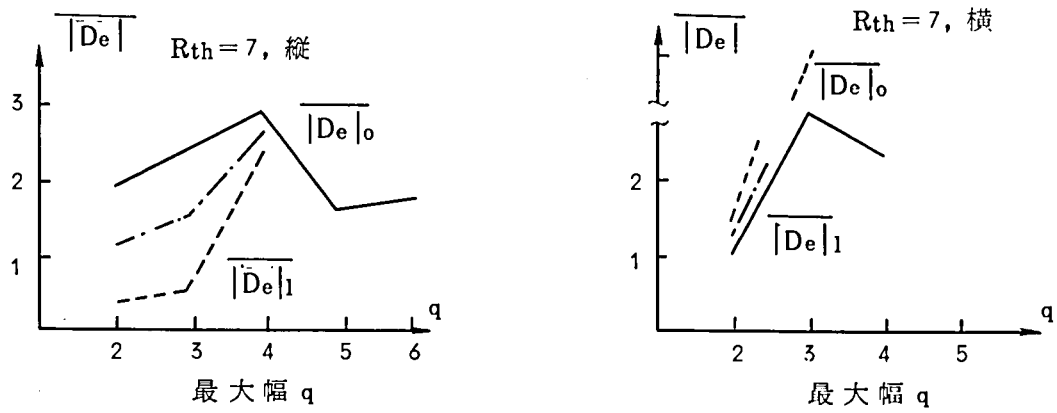
横ラン群の傾きも同様に求まる。

以上で定義したラン群の傾きをサンプルパターンについて求め，その絶対値の類別平均と最大幅 q の関係を図 5.11 (1) に示す。



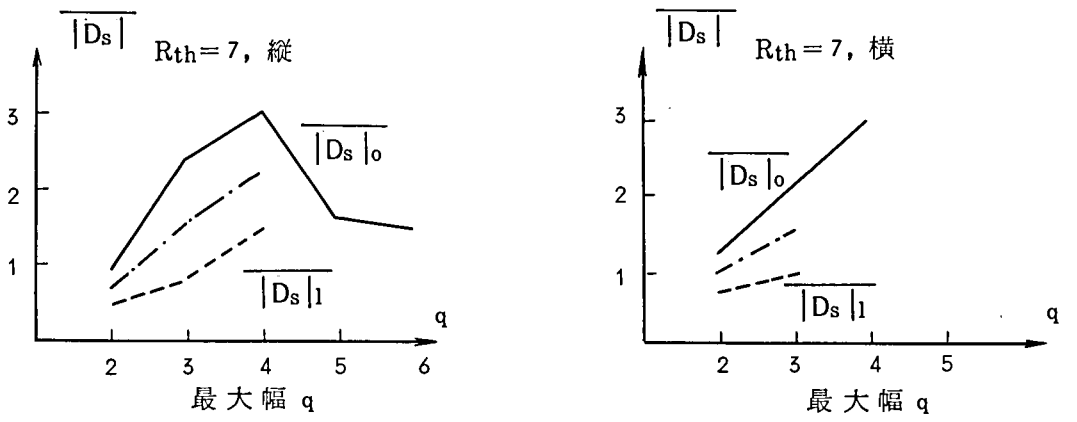
(a) 縦 ラ ン 群　　(b) 横 ラ ン 群

図 5.11—(1)　ラン群の最大幅 q と傾き A の関係



(a) 縦 ラ ン 群　　(b) 横 ラ ン 群

図 5.11—(2)　最大幅 q と始点最大差 $|D_e|$ の関係



(a) 縦 ラ ン 群　　(b) 横 ラ ン 群

図 5.11—(3)　最大幅 q と終点最大差 $|D_s|$ の関係

② 始点最大差と終点最大差

式（5.5）で表わされる縦ラン群に対して，ラン群の始点最大差 D_s と終点最大差 D_e を次で定義する。

$$D_s = max\{i_{s,k}\} - min\{i_{s,k}\} \qquad (5.17)$$

$$D_e = max\{i_{e,k}\} - min\{i_{e,k}\} \qquad (5.18)$$

$$(1 \leqq k \leqq r)$$

横ラン群についても同様に定義する。

図 5.8 に示した縦ラン群では，始点最大差 D_s と終点最大差 D_e はそれぞれ 2 と 1 である。

図 5.11(2)，(3)に終点最大差と始点最大差の類別平均と最大幅 q の関係を示している。

(iii) ラン群の 2 次類別

2 次類別に当っては，前述の複数個のパラメータを組み合わせて，類別の閾値をラン群幅の最大値 q の関数として使用するのが高い類別精度を与えると思われる。計算機による類別処理時間を短縮するため 1 種類のパラメータだけを使用するときは，類別のための閾値としては，図 5.11 に 1 点鎖線で示す各パラメータの類別平均値の平均値を用いればよい。

例えば，ラン群の傾き A を使用して類別するためには，まず式（5.13）～（5.16）を用いてラン群の傾きを調べ，これが閾値以上なら o ラン群，以下なら l ラ

ン群と類別する。

図5.11の結果によると縦ラン群の類別には傾きA_vを用いるのが最も完全に近い結果を与える。横ラン群の類別は縦ラン群の類別に比較してより困難であるが，これはサンプルパターンが明朝体であるため，横線が細く，かつ，いわゆる“ウロコ”が線端に存在するためである。

なお，最大幅1の横ラン群については特徴パラメータによる2次分類は不可能であるが，図5.10で分かるように，それらが大部分lラン群であることからlラン群に類別することとした。

(iv) 縦横直線の分離精度

以上述べた類別手法をサンプルパターンに適用し，縦横直線の分離精度を求めた。分離が最も困難な場合は「力」，「解」などに含まれる縦横直線に近い斜線，曲線と真の縦横直線の分離の場合である。よって，分離精度も実験に用いるサンプルパターンに依存する。

本実験で用いたサンプルパターンは縦横直線だけから構成されるもの32文字，縦横直線に近い斜線，曲線を含むもの30字の計94文字から構成している。

サンプルパターンに本分離法を適用したときの正しく分離された直線数と，誤まって分離された直線数をそれぞれ図5.12の左方向と右方向に，パターン中の全直線数で正規化して示している。同図(a)は最大幅による1次類別とラン群の傾きによる2次類別を行なった場合，同図(b)は1次，2次類別とも行なわなかった場合である。

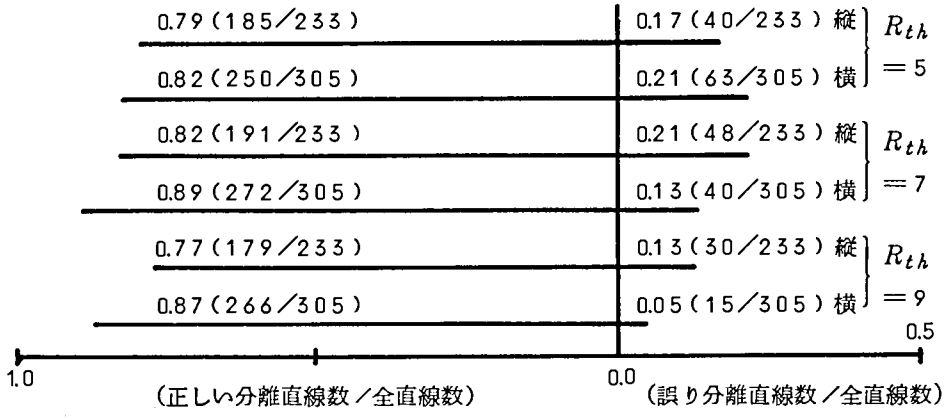
誤まって分離される直線数と正しく分離される直線数は，類別を行なうことにより両者とも減少するが，両者の比（誤まって分離される直線数／正しく分離される直線数）は，$R_{th}=9$のとき類別を行なった場合に最小となる。

誤まって分離される直線数を，類別処理を行なう場合と，類別処理をしない場合で比較し，類別処理の効果を見ると縦直線に対する効果が横直線に対するそれよりも大きく，ラン群の2次類別に用いた特徴パラメータが縦ラン群向きであったことが分かる。

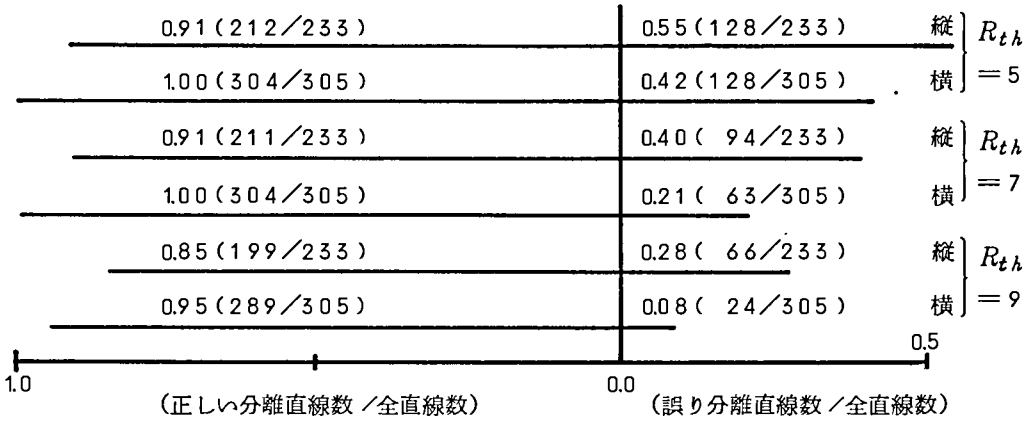
なお，それにもかかわらず，$R_{th}=7$，9の場合，1次，2次の類別処理を行なっても誤まって分離された縦直線数が横直線数よりも多いのは，サンプルパターン中に縦直線に近い斜線，曲線が横直線に近いそれよりも多く含まれていたた

めであると考えられる。

以上の検討結果，ラン群の類別処理は，正しい直線の分離率を少し下げるが，整形処理後の漢字パターンの文字品質を劣化させ，人間による修正作業を困難にする，誤分離直線数を減らす上で有効であるといえる。



(a)　1 次類別と傾きによる 2 次類別を行った場合



(b)　1 次類別，2 次類別とも行わなかった場合

図 5.1 2　サンプルパターンについての縦横直線分離割合

5.3.4　縦横直線処理

前項で述べた縦横直線と斜線，曲線の分離処理の結果，l ラン群として類別されたラン群から縦横直線を抽出する。

式（5.5）で表わされる縦ラン群の場合には次式の直線を縦直線として抽出する。

$$\{ (\min_k (i_{s,k}), \min_k (j_k)), (\max_k (i_{e,k}) + \triangle, \min_k (j_k)) \}$$

△；終点補正　　（5.19）

min, max はそれぞれ最大値，最小値をとることを示す。また，min, max をとる対象のランは，当該ラン群中の最大のラン長の半分以上のラン長をもつランとした。これは斜線，曲線から誤まって抽出された直線に対しても，傾きあるいは曲率の大きな部分の影響を除くためである。

式（5.19）はラン群の左端に，ラン群を i 軸上に投影した時に生じる影の長さをもつ縦直線を抽出することを示している。したがって，発生させる直線は抽出した直線の右側に幅をもつこととする。このようにラン群の左端に直線を発生させる理由は，縦直線の下端にしばしば存在する短い横線である“はね”（これは通常，横直線として抽出されないため位置の制御は出来ない）と縦直線の接続を行なうためである。

終点補正△は原漢字パターンと整形後の漢字パターンの線幅が異なる場合に必要となる。

図5.13に示す，線幅3の原漢字パターン「ロ」を線幅2の整形漢字パターンとして発生させる場合を例に考察する。

この例では，次の4本の縦，横直線

直線 H_1 ； $\{(i_1, j_1), (i_1, j_3)\}$

直線 H_2 ； $\{(i_2, j_1), (i_2, j_3)\}$

直線 V_1 ； $\{(i_1, j_1), (i_3, j_1)\}$

直線 V_2 ； $\{(i_1, j_2), (i_3, j_2)\}$

が抽出される。

これらの直線情報を基に線幅2の直線を発生させると，各直線の終端に計8画素のノッチが新たに発生することが分かる。

終点補正△は，これらのノッチの発生を抑えるためのものであり，縦直線の終点補正 $\triangle_v$ は次式で与えられる。

$$\triangle_v = W_{hp} - W_{ho} \qquad (5.20)$$

但し，W_{ho}, W_{hp} はそれぞれ原漢字パターンと整数漢字パターンの横線幅である。

以上，縦直線について述べてきたが横直線に対しても同様である。

分離された縦横直線情報を基に所定の線幅の縦横直線を発生させることにより，線幅が統一された縦横直線部が作成される。この例を図5.9(d)に示している。

なお，発生すべき直線は全て縦，横直線であるから，特別な発生アルゴリズムを用

いる必要はない。

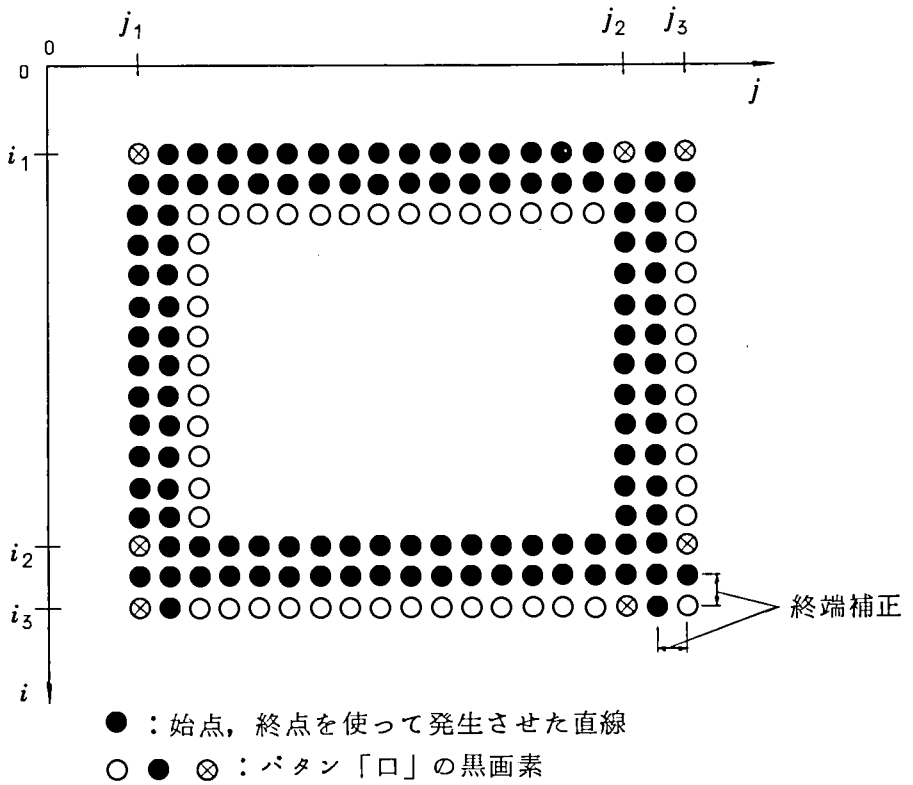


図 5.1 3　「口」のパターンと終端補正

5.3.5　斜線，曲線部処理

縦横直線分離処理で l ラン群に類別されたラン群を構成するランで，そのラン長が当該ラン群内の最大ラン長の１／２ 以上のランを原漢字パターンから消去する。この結果得られるパターンを図 5.9 (c) に示す。

原漢字パターンのノッチは，このとき，幅１画素の画素列としてこのパターン内に残される。そこで，次式のノッチ除去フィルタを用いて，幅１の画素列を消去する処理を行ないノッチを除去する。

すなわち，黒画素を真，白画素を偽とするとき

$$\overline{a(i-1,\ j)} \otimes \overline{a(i+1,\ j)} \oplus \overline{a(i,\ j-1)} \otimes \overline{a(i,\ j+1)}$$

⊗ ；論理積

⊕ ；論理和

—— ；否　定　　(5.2 1)

が真であれば，画素$a(i, j)$を白画素とする。

ノッチを除去した結果を図5.9(e)に示す。これをノッチを除去した斜線曲線部とする。

最後に，線幅が統一された縦横直線部とノッチが除去された斜線，曲線部を再合成することにより整形処理結果の漢字パターンを得る。この例を図5.9(f)に示す。

5.3.6 整形処理の効果

コード表（社内漢字コード表）に従って，連続する１００個の漢字を対象に，（３２×３２）のゴシック体漢字パターンを作る場合について整形処理の効果を調べた。

先ず，原漢字パターンを基に人間（素人）が自由に（但し，パターンの平行移動は禁止して）画素配列の修正を行ない完成漢字パターンを作成し，得られた完成漢字パターンと原漢字パターンとで状態の異なる画素数を基本修正点数として求める。

次に，原漢字パターンに対して整形処理を施して整形漢字パターンを作成し，整形漢字パターンと完成漢字パターンの差の画素数を改善修正点数として求め，次式に定義する整形効率ηを求めた。

$$\eta = \frac{\text{基本修正点数} - \text{改善修正点数}}{\text{基本修正点数}} \qquad (5.22)$$

前記の１００字に対して，基本修正点数が3,879，改善修正点数が2,603となり，整形効率0.33が得られた。

すなわち，整形処理により修正点数は約２／３に削減された。

図5.14に整形処理の観点からみた特徴的な漢字パターンについて，原漢字パターン，完成漢字パターンおよび基本修正点と改善修正点の分布を示している。なお，図で，整形処理を行なうことによって新たに生じる修正点を改善修正点分布図の中で•印で示している。

これらの漢字パターンを含めて，パターン別に，整形処理の効果を評価すると次のようになる。

(1) 整形処理の効果が顕著なもの

図5.14で「問」，「喜」，「匿」，「回」のように，整形処理の内容から，当然縦横直線を主たる構成要素とする漢字パターンに対して整形処理の効果は著しい。

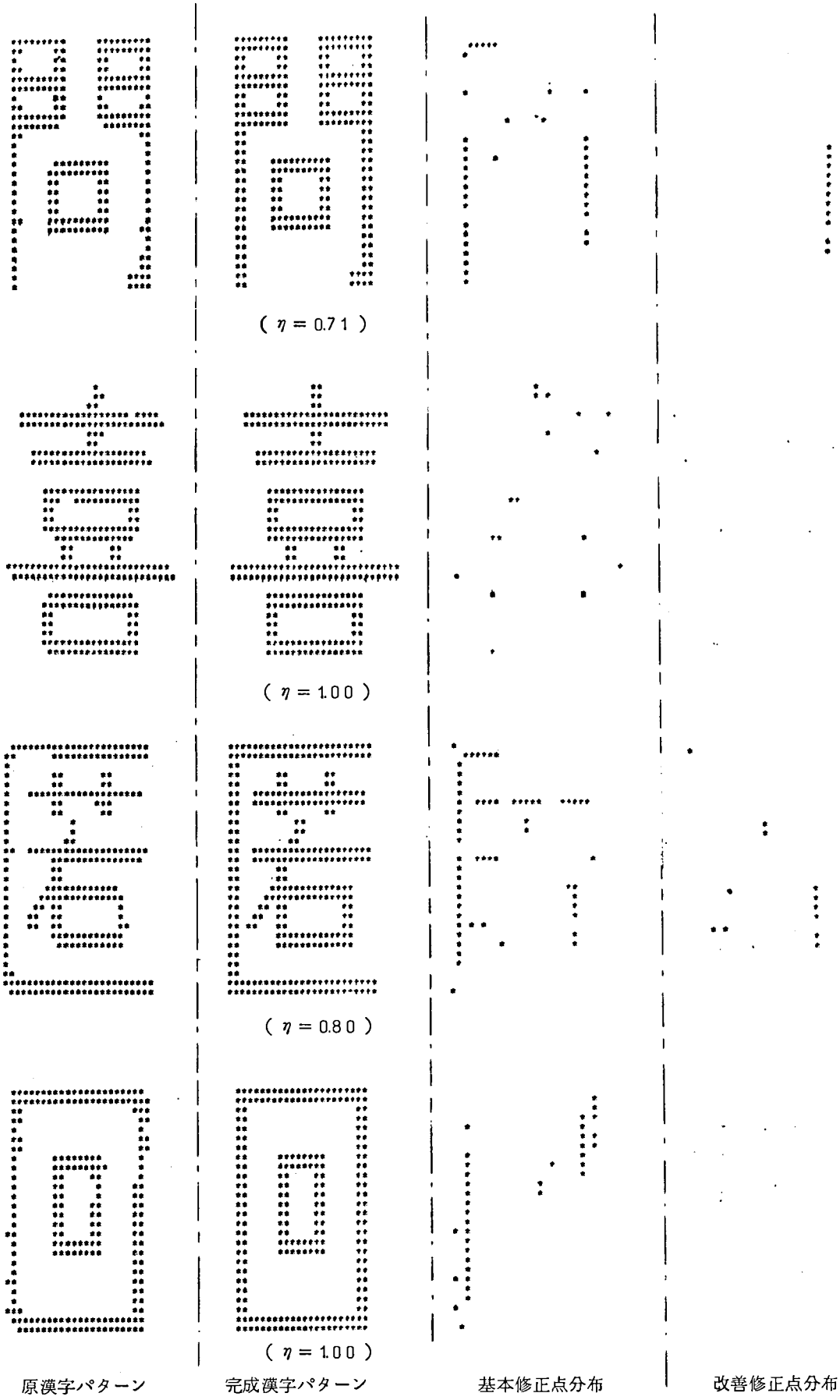


図 5. 1 4　修正点分布の例(1)

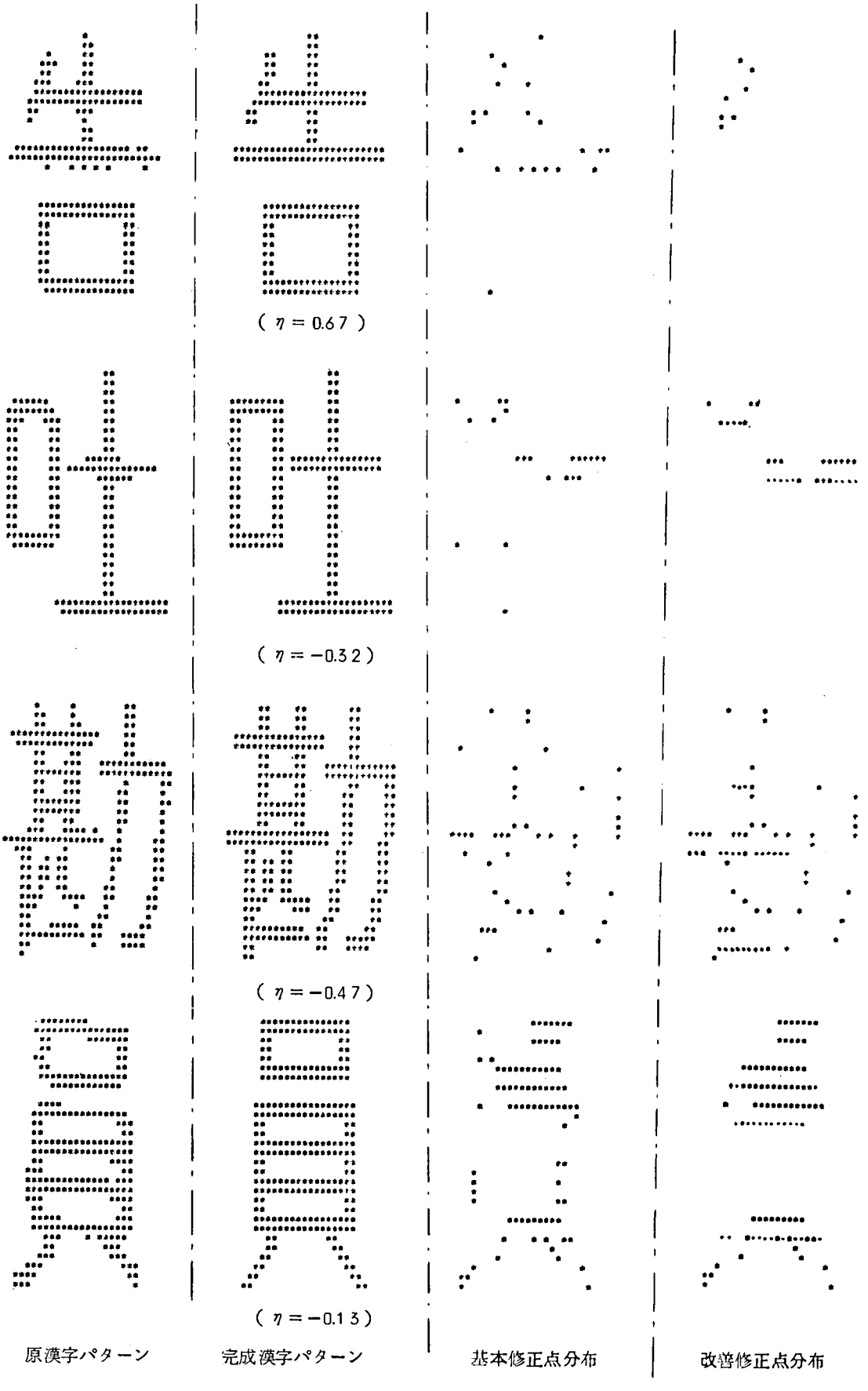


図 5. 1 4　修正点分布の例(2)

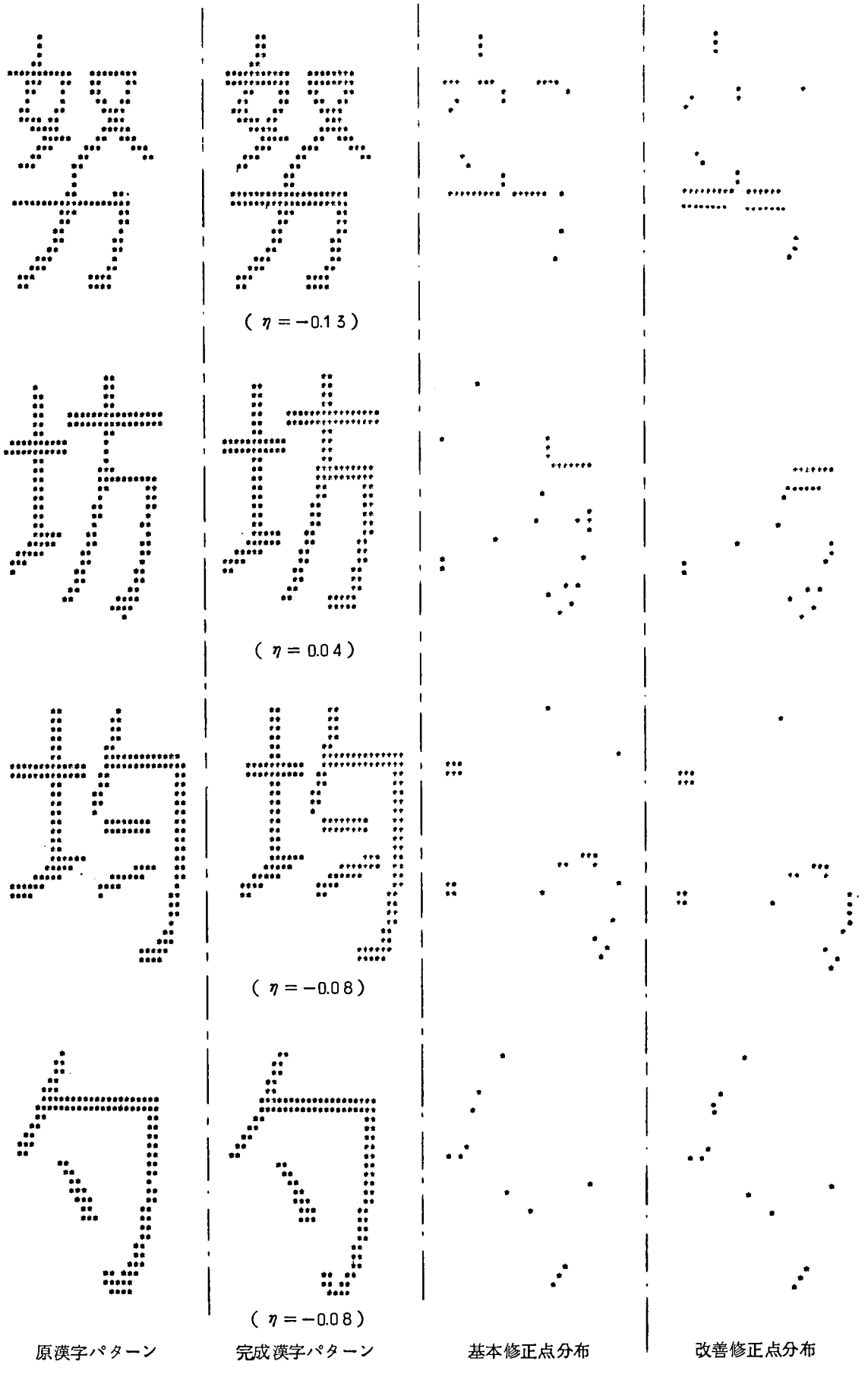


図 5. 1 4　修正点分布の例(3)

(2) 整形処理の効果がないもの

整形処理の効果が上がらない要因としては次の2つがある。

① 縦横直線の位置ずれ

縦横直線の位置ずれは大幅な修正点の増加をもたらす。したがって，「吐」，「勘」，「員」のように縦横直線を多数含むパターンであっても直線の位置の修正を要する場合には整形処理の効果は見られず，逆に，修正点数の増加をもたらす場合がある。

これは，5.3.4項で述べたように，抽出した直線の線幅を縦直線については右側に，横直線については下側に，一意的に拡大しているためであり，これを直線の両側で黒画素数の多い方に線幅を拡大することによって改善することが考えられる。

この方法により，図5.14に示した漢字パターンの例では整形効率ηは，「吐」，「勘」，「坊」について，それぞれ−0.05（−0.32），0.33（−0.47），0.52（0.05）へと大幅に向上する。但し，（　）の中は現状の整形処理による整形効率である。

② 斜線・曲線要素

斜線・曲線要素に対する整形効果は無い。したがって，「努」，「均」，「勺」のような斜線・曲線を主要な構成要素とする漢字パターンに対して整形効果は見られない。

5.3.7 サイズ変換，書体変換機能

以上，本装置ではサイズ（32×32）の漢字パターンを標準サイズの漢字パターンとして作成処理を行なうが，標準サイズ以外の漢字パターンの作成に対してはサイズ変換機能を，漢字母形と異なる書体の作成に対しては書体変換機能をもっている。

これらの機能は5.3.3項で述べた縦横直線部の処理に容易に追加することができる。

(1) サイズ変換処理

縦横直線と斜線，曲線が分離された後では，縦横直線に対してはストローク式漢字パターンに対するサイズ変換法を，斜線，曲線に対しては3章で述べた比例法によるサイズ変換を独立に適用することになる。

すなわち，原漢字パターンから分離された縦横直線の始点，終点座標

$$\{(i_1, j_1), (i_2, j_2)\} \qquad (5.23)$$

と斜線，曲線部の黒画素の座標

$$(i_3, j_3) \qquad (5.24)$$

に対して，次式に示す同一の座標変換を行ない，$\{(I_1, J_1), (I_2, J_2)\}, (I_3, J_3)$ を得る。

$$I_* = \left[\frac{M-1}{m-1} \cdot (i_* - 1) + 1.5\right] \qquad (5.25)$$

$$J_* = \left[\frac{N-1}{n-1} \cdot (j_* - 1) + 1.5\right] \qquad (5.26)$$

但し，＊は１，２，３を表わし，（$m \times n$）は原漢字パターンのサイズ，（$M \times N$）は変換後の漢字パターンのサイズ，〔　〕はガウス記号である。

変換された始点，終点の座標を用いて縦横直線を発生させ，変換後の斜線，曲線と合成することにより，線幅が統一され，かつ，文字バランスの劣化のないサイズ変換漢字パターンが得られる。

図 5.１５に印刷漢字母形を基にしたサイズ変換結果の例を示す。

(2)　書体変換処理

図 5.１６にゴシック体から明朝体へ書体変換した例を示す。

書体変換処理の基本的な考え方は第４章で述べた書体変換処理の場合と同じである。

図 5.１６に示すゴシック体から明朝体への変換処理では式（ 5.１９ ）として抽出された縦直線

$$l_v ; \{(i_{sv}, j_v), (i_{ev}, j_v)\} \qquad (5.27)$$

$$v = 1, 2, \cdots, V$$

と同様にして抽出された横直線

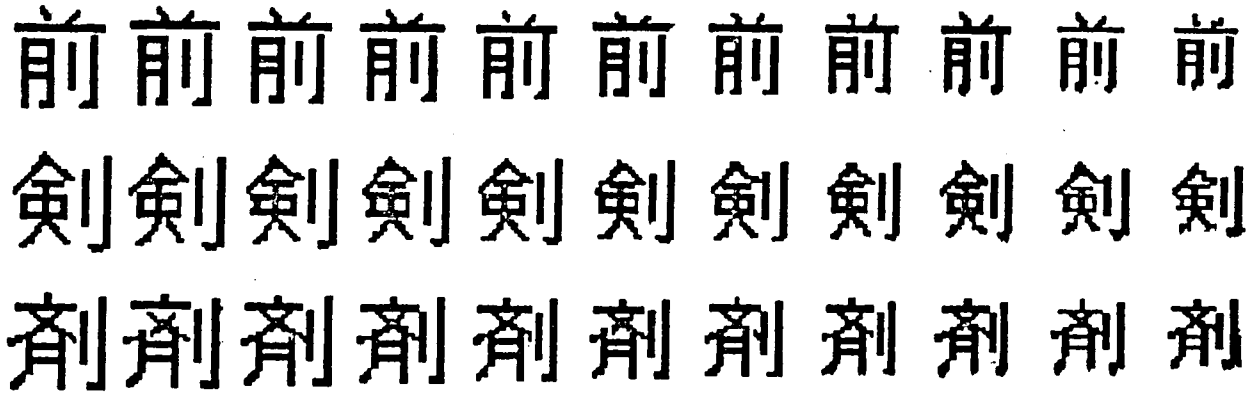


図 5.１５　サイズ変換の例（（３２×３２）から（２０×２０）への縮小変換）

（原ゴシック体漢字パターン）

（擬似明朝体漢字パターン）

図 5.16　書体変換の例

$$l_h\ ;\ \{\ (\ i_h,\ j_{sh}\ ),\ (i_h,\ j_{eh}\ )\}\qquad(5.28)$$

$$h=1,\ 2,\ \cdots,\ H$$

に対して次の処理を行なう。ただし，式（5.27），（5.28）でV，Hはそれぞれ漢字パターン内で抽出された縦直線と横直線の個数である。

(i) 横直線は式（5.28）で与えられる線幅1画素の直線を発生する。

(ii) 縦直線は式（5.27）で与えられる直線を左側とする線幅2画素の縦直線を発生する。

(iii) 式（5.27），（5.28）で与えられる縦，横直線の線端座標を用いて，4章で述べた条件で線端飾りの付加位置と種類の識別を行ない，線端飾りを付加する。

(3) サイズ・書体変換処理

縦，横直線を両端を含めて完全に抽出した結果としてサイズ変換と書体変換の両処理を容易に組み合わせることができる。

すなわち，

(i) 縦，横直線の両端の座標を式（5.25），（5.26）に従って座標変換する。

(ii) 変換された縦，横直線の両端の座標を用いて線端飾りを付加する。

(iii) 斜線，曲線部の黒画素は式（5.25），（5.26）に従って座標変換する。

以上の処理によって得られたゴシック体から明朝体へのサイズ・書体変換処理結果の例を図5.17に示す。

この変換では線端飾りの形状をサイズにかかわらず固定しているため，サイズが小さくなると不自然さが目立ってくる。今後，サイズに応じて線端飾りの形状を変える

改善法が考えられる。

向向向向向向向向向向

向向向向向向向向向向

サイズ変換ゴシック体

向向向向向向向向向向

向向向向向向向向向向

サイズ書体変換明朝体

君君君君君君君君君君

君君君君君君君君君君

サイズ変換ゴシック体

君君君君君君君君君君

君君君君君君君君君君

サイズ書体変換明朝体

味味味味味味味味味味

味味味味味味味味味味

サイズ変換ゴシック体

味味味味味味味味味味

味味味味味味味味味味

サイズ書体変換明朝体

32	31	30	29	28	27	26	25	24	22	21

図 5.17　サイズ・書体変換例

5.3.8　プログラムの構成

整形処理プログラムの規模を表5.1にスタティックステップ数で示している。

表5.1　整形処理プログラムの規模

プログラム名	処　理　内　容	ステップ数
REFINE	整形処理全体を制御する	45
SEP	縦横直線と斜線曲線を分離する	60
WIDTH	線幅を規格化する	131
NOTCH	ノッチを除去する	56
RECON	直線を発生させる	29

整形処理プログラムは漢字パターン作成処理プログラムの中で1つのサブルーチン化されており，さらに，表5.1に示すような個々の機能別のサブルーチンをCALLする構成をとっている。プログラムの構成を図5.18に示している。

なお，使用言語は全てフォートランである。

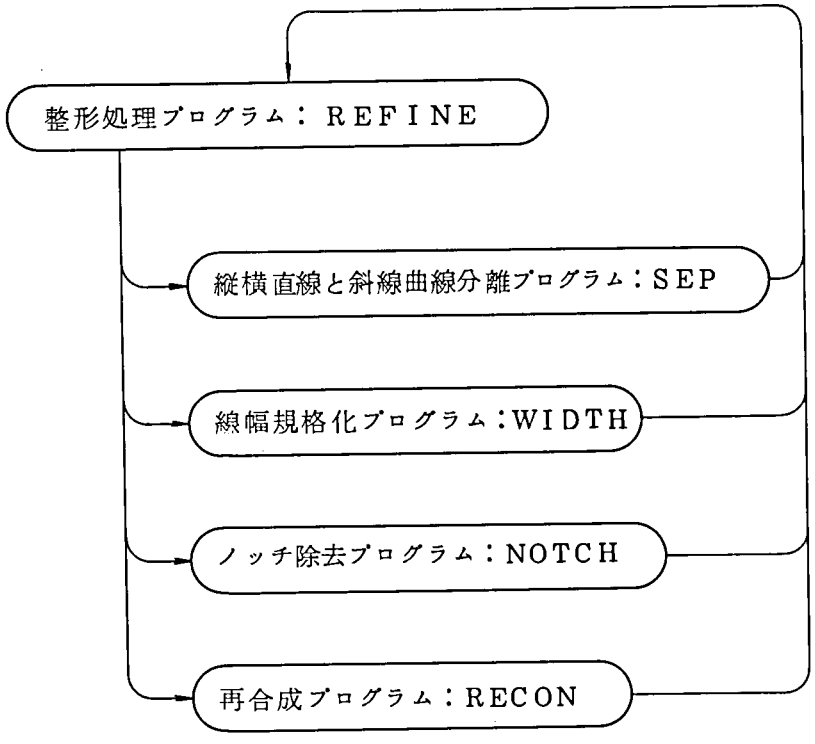


図5.18　整形処理プログラムの構成

5.4 漢字パターンの作成

本漢字パターン作成装置を用いてサイズ（３２×３２）の明朝体とゴシック体の漢字パターンを作成した。

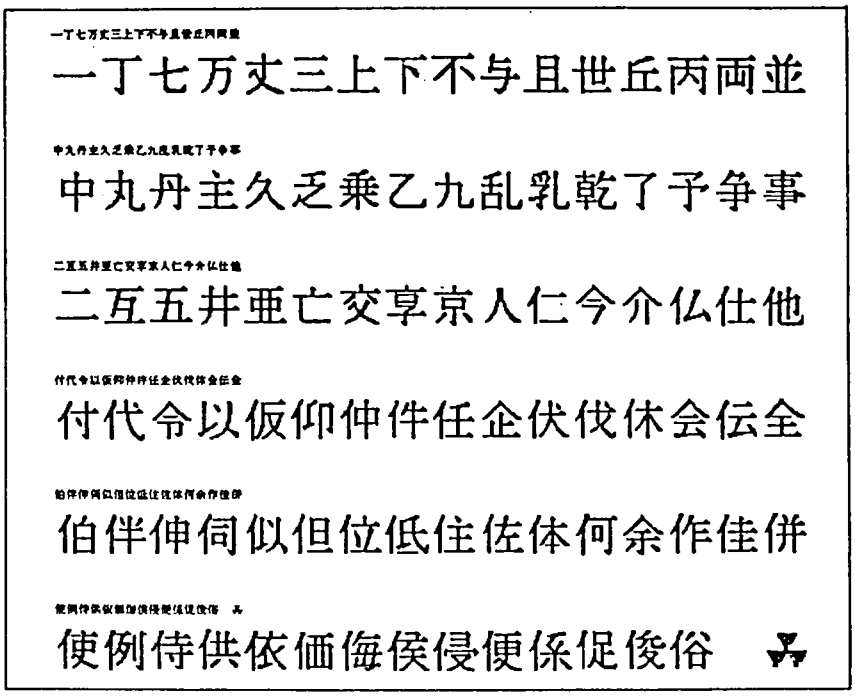
2,９００字の明朝体漢字パターン作成実験を通して得られた，作成時間，会話修正時間，修正点数を表5.2に示す。

また作成した漢字パターンの例を図5.１９に示す。

この結果，（３２×３２）の漢字パターン１文字を平均約2.4分で作成することが出来，漢字パターンの効率的な作成という目的が達成された。

表5.2　漢字パターンの作成実験結果

測　定　項　目	測　定　値
平均作成時間（秒／字）	１４５
画素単位の会話修正時間（秒／字）	９３
平均修正画素数（画素／字）	７９



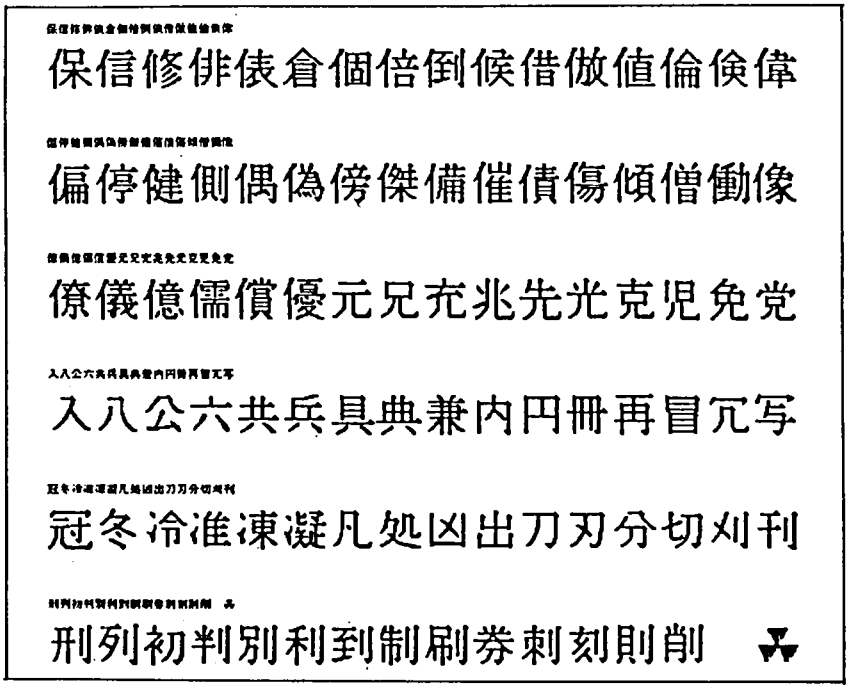
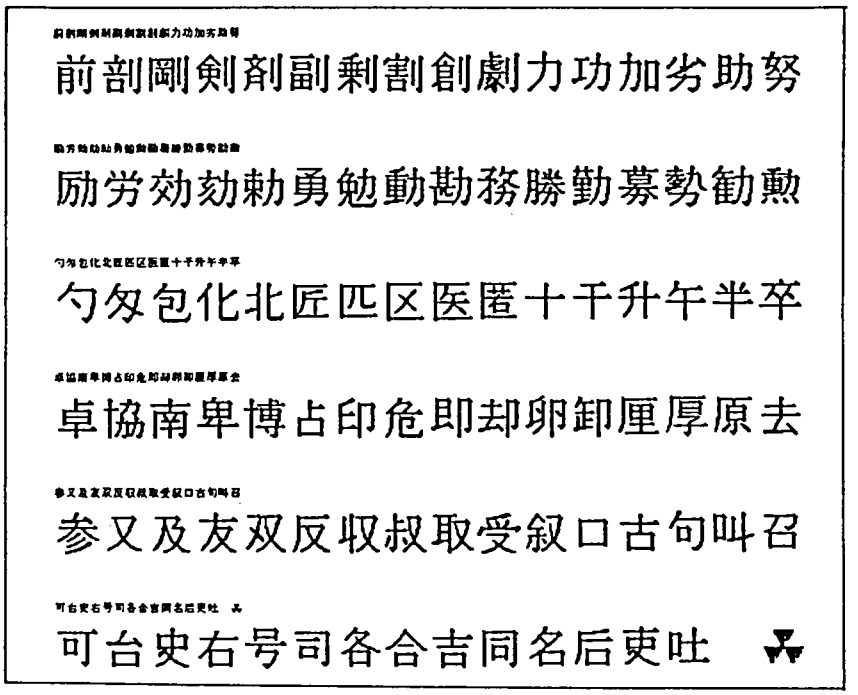


図 5.19　作成した漢字パターンの例（３２×３２）（その１）



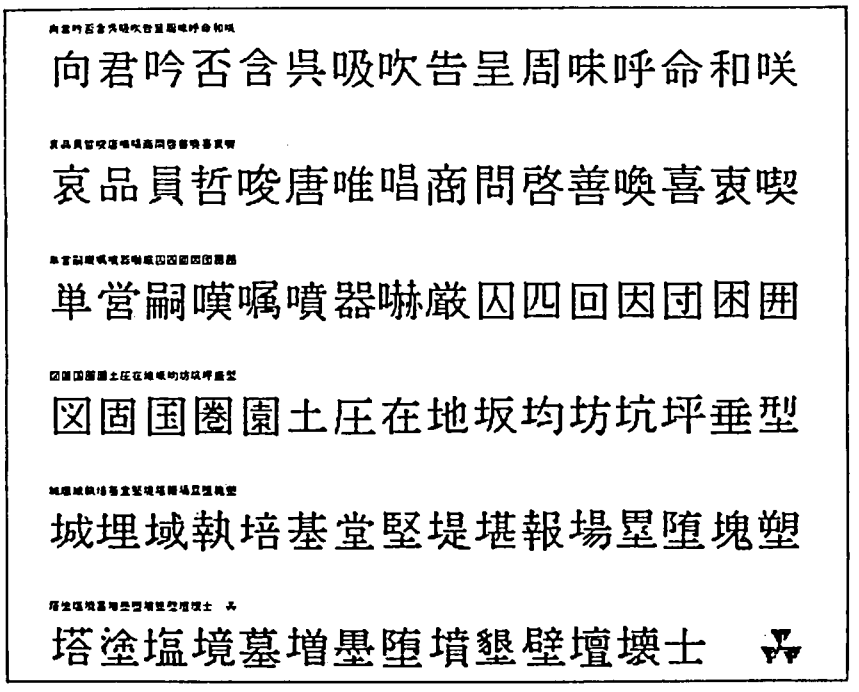


図5.19　作成した漢字パターンの例（その２）

5.5 ま と め

従来，画素形漢字パターンの作成は人手で行なわれており，多大の労力，時間それに経費を必要とした。

これの問題を解決する技術として，ディスプレイ装置を介した計算機と人間との会話処理による効率的な漢字パターン作成装置に関する研究が本研究に先立って，いくつか報告されていた。

本研究では計算機と人間との会話処理という基本的な考え方は従来の研究結果を踏襲し，さらに，人間の労力を軽減することを目的とした自動整形処理について考察した。

その結果，

(1) 縦横直線と斜線，曲線の分離処理に基づく，縦横直線の線幅の統一とノッチの除去を可能とする整形処理手法を提案した。

(2) 互いに隣接する黒ランをまとめてラン群とし，ラン群がもつ特徴パラメータによる縦横直線と斜線曲線の分離手法を提案し，各特徴パラメータによる分離能力を考察し，ラン群とラン群の傾きが分離パラメータとして有効なことを示した。

(3) 整形処理の効果として，ゴシック体の漢字パターン作成時に，整形処理を行なうことにより，そうでない場合に比較して，修正点数を約２／３に削減できることを示した。

(4) 整形処理を効果的に機能させるための漢字パターン作成装置の構成方式として，計算機による同一漢字パターンに対する複数処理結果の中から人間が最適結果を選択する会話処理方式とすることにより，従来の装置に比較して人間と機械の機能分担をより明確とした。

この結果，（３２×３２）の明朝体漢字パターン１個を平均約2.4分で作成することができ，作成効率の向上が図られた。

なお，今後，整形処理漢字パターンと人手による完成漢字パターンとの間の縦横直線の位置ずれに基づく修正点数の増加を防ぐために，縦横直線幅の拡大方向の決定法に対する改善が必要である。

付録5.1　装置構成機器の処理分担と主な仕様

構成装置	主な処理内容	主な仕様
FSS	• カード上に記録された漢字母形の読み取り	走査範囲；96㎜×72㎜ 走査線数；512～4,096 速度｛2値；16μs／画素 多値；112μs／画素
GDP	• 作成処理途上の各種漢字パターンの表示と会話処理	表示域；14″×13″ 格子数；4096×4096 輝度レベル；32 表示速度；18,000リニアインチ／30fPS
小型計算機	• 装置全体の制御 • FSS，GDPの入力装置の制御 • 漢字パターンの整形処理 • 漢字パターンの格納・検索制御	語長；16ビット メモリ容量；32K語 平均命令実行時間；1.7～2.5μs
補助記憶装置	• 読み取り，修正用，完成の各漢字パターンの格納 • 各プログラムの格納	磁気ディスク装置；1.2M語 磁気テープ装置；1600bpi
ドットプリンタ	• 完成漢字パターンのハードコピー出力	サイズ；B4 解像度；8本／㎜

付録5.2　実験に使用したサンプルパターン94文字

一丁三上且世両中事亜円冊再冒出刊副十卓博占口古
司可吉同吐呈品唱問　　不並了交京仁企休全余侵俗
六共兼冬凍凶合否味困在基堅塁墓奇宗客対岐
石乏倉倣冗刀剤力助勝危卵厘双叫后吏託吸周声夫妃
尼局屋川巡帰幻

付録5.3　式（5.13）の導出

縦ラン群

$$\{(i_{sk}, j_k), (i_{ek}, j_k)\} \qquad (A.1)$$

$$(k=1, 2, \cdots, r)$$

を構成する黒画素の全位置を

$$\{(i_d, j_d)\} \qquad (d=1, 2, \cdots, D) \qquad (A.2)$$

とする。

直線

$$j = A_v \cdot i + B_v \qquad (A.3)$$

とラン群を構成する各黒画素との j 軸方向への距離の2乗和は，

$$T=\sum_{d=1}^{D}\{ j_d-(A_v \cdot i_d+B_v)\}^2 \qquad (A.4)$$

となる。

式（A.4）の2乗和 T を最小とする A_v は，式（A.4）を A_v，B_v で偏微分し，それを零とおくことにより，次のように求まる。

$$A_v=\frac{\sum_d i_d \cdot j_d-\sum_d i_d \cdot \sum_d j_d/S}{\sum_d i_d^2-(\sum_d i_d)^2/S} \qquad (A.5)$$

$$(d=1, 2, \cdots, D)$$

さらに，式（A.5）の中の $\Sigma i_d \cdot j_d$，Σi_d，Σj_d，Σi_d^2，S は，式（A.1）のラン群の端点座標を用いて次のように表わされる。

$$\sum_d i_d \cdot j_d=\sum_k j_k \cdot (i_{sk}+i_{ek}) \cdot (i_{ek}-i_{sk}+1)/2 \qquad (A.6)$$

$$\sum_d i_d=\sum_k (i_{sk}+i_{ek}) \cdot (i_{ek}-i_{sk}+1)/2 \qquad (A.7)$$

$$\sum_d j_d=\sum_k j_k \cdot (i_{ek}-i_{sk}+1) \qquad (A.8)$$

$$\sum_d i_d^2=\sum_k \{ i_{ek} \cdot (i_{ek}+1)(2 \cdot i_{ek}+1) - (i_{sk}-1) \cdot i_{sk} \cdot (2 i_{sk}-1)\}/6 \qquad (A.9)$$

$$S=\sum_k (i_{ek}-i_{sk}+1) \qquad (A.10)$$

式（A.6）～（A.10）を式（A.5）に代入して整理すると式（5.13）を得る。

第6章　結　　論

本論文では画素形漢字パターンの出力処理に関する研究を述べたが，以下に本研究で得られた成果を要約して記す。また残された問題点についても言及する。

第2章では，複数台のラスタ走査形出力装置で共用される漢字パターン発生器への適用をねらった行パターン合成符号化法と画素形漢字パターンのサイズとデータ圧縮比との定量的な関係について考察した。

漢字パターンのデータ圧縮法はデータ圧縮効率とともに，復号処理に伴なう発生速度の低下，装置規模も考慮して漢字パターン発生器の適用システム対応に評価しなければならない。

本研究では高価な漢字パターン発生器を多数のラスタ走査形出力装置で共用するシステムへ適用することを前提として，漢字パターンを出力装置の走査方向と方向が一致する1次元部分パターン（行パターン）へと分割し符号化する行パターン合成符号化法を提案し，その圧縮効率と発生速度について検討した。

その結果，（18×16），（24×24），（32×32）の各字パターンに対してそれぞれ77%，66%，49%にデータ量が圧縮されることを示した。代表的な1次元符号化法であるランレングス符号化法と比較して，データ圧縮率で5～11%高い圧縮効果をもち，漢字パターンの発生速度も数倍速いことを示した。

また，従来の部分パターンへの分割符号化法の検討では部分パターンの大きさの最適化に対する考察はなされていなかったが，部分パターン自体の表現に要するデータ量と部分パターンの符号化データ量の両者を考慮することによって，データ圧縮効果を最大とする部分パターン（行パターン）の大きさの最適値があることを示した。

次に，漢字パターンのサイズとデータ圧縮比に対する考察では，1次元，2次元のそれぞれの符号化法のモデルを設定し，各符号化法に対して，サイズとデータ圧縮比の関係が，サイズ（24×24）～（64×64）の範囲で，1次式で近似できることを示した。特に，（24×24），（32×32），（48×48），（64×64）のサイズで作成された40個の漢字パターンを基に，定量的なデータ圧縮比の予測式を示した。

また，漢字パターンのサイズの変化に伴なうラン当りのエントロピーの変化について考察し，黒ランについては単純マルコフ過程に従うランの発生モデルは漢字パターンに対しては精度良く成立しないことを指摘した。

第3章では画素形漢字パターンのサイズ変換法について述べた。

従来，画素形漢字パターンのサイズ変換法としては整数倍の拡大変換法が見られた。しかし，それらの方法ではサイズ変換を行なうために必要な画素の挿入，削除を個々の漢字パターンの形状に無関係に同一ケ所に対して行なう方法であったため，字画の欠落や線幅の不揃いを避けるためには整数倍の拡大変換に機能を限定せざるを得なかった。

本研究では挿入，削除する画素を漢字パターン対応に選択するという考え方を新たに導入することによって，縮小変換まで含めて非整数倍のサイズ変換を可能とした。

すなわち，新たに，挿入・削除画素の選択のために，サイズ変換に対する要求条件と画素の選択基準を設定し，文字品質の主要な支配要因である縦横直線の線幅と文字バランスの両者の制御が可能である全自動のサイズ変換アルゴリズムとして線分の比例配置法を提案した。

さらに，上記の漢字パターン対応に挿入・削除画素を選択するという考え方を，変換速度，変換文字品質の両面で実用可能なレベルで具体化した方法として，行列および画素の選択情報を漢字パターン対応に記憶する制御情報付与方式を提案した。

制御情報を付与するサイズ変換法は漢字パターンのデータ圧縮法としても位置づけることが出来，その時，データ量は（１８×１６）の漢字パターンに対して1/4～1/7程度に削減できることを示した。

第4章では画素形漢字パターンの書体変換法について述べた。

我が国の代表的な漢字の書体である明朝体とゴシック体の両書体間の主要な差異は①縦，横直線の線幅比，②縦横直線の線端における線端飾りの有無，それに③斜線，曲線の形状にある。本研究では縦，横直線の線幅比と線端飾りを対象にその制御法を考察した。

その結果，ランの抽出処理を基本とした比較的簡単な処理で線幅比と線端飾りの制御が可能であることを示した。

また，人手で作成された漢字パターンを原漢字パターンとして，原漢字パターンと書体変換パターンとの差分パターンの黒画素数を用いて書体変換の達成度を評価し，書体変換処理により，原漢字パターンでの両書体間の差分黒画素数で換算して，約7割の書体変換の達成率が得られることを示した。

第5章では漢字パターン作成処理として，主に，漢字パターンの自動整形処理について述べた。

漢字母形をディジタル化する時，線分周辺のノッチと線幅の不揃いの発生は避けられず会話処理による漢字パターンの作成では，これらノッチの除去と線幅の統一にパターン作成労力の

多くが費やされる。

そこで，ノッチの除去と縦横直線の線幅の統一を自動整形処理の目的として検討し，縦横直線と斜線・曲線の分離処理が有効であることを示した。

特に，縦横直線の抽出法として，互いに隣接する黒ランをまとめてラン群とし，ラン群が持つ特徴パラメータによる縦横直線と斜線曲線の分離手法を提案し，各特徴パラメータの分離能力について考察した。その結果，ラン群の幅と傾きの組み合わせにより高い分離精度を与えることを示した。

また，自動整形処理を有効に機能させるための会話処理方式についても述べた。

以上，各章を処理の目的別に分けて述べたが，処理の内容から見ると第３章から第５章の範囲は全て，漢字パターンの縦・横直線と斜線，曲線との分離処理を基本としている。

縦横直線と斜線・曲線の完全な分離は困難であるが，本研究では，完全な分離精度が得られなくても，いくつかの処理が可能であることを示した。

第３章のサイズ変換処理では，縦横直線は挿入削除する行および列の選択のための重みとして抽出できれば良く，行，列内でのその位置を求めることなく，限定された範囲でのサイズ変換が可能であることを示した。また，第４章の書体変換処理における線端飾りの制御では，隣接する２本の黒ランの端点が一致する点のみを縦横直線情報として用いれば線端飾りの制御が可能であることを示した。

第５章では未整形漢字パターンを対象とする点で第３章，第４章の処理と基本的に異なっており，縦横直線の形状そのものを変形する必要から，初めて，両端を含めて陽に縦横直線を抽出する処理について言及した。縦横直線が陽に抽出できれば，当然，サイズ変換，書体変換は可能であり，その１例を５章で示した。

今後に残された課題としては，サイズ変換，書体変換，整形処理を通して，斜線・曲線部に対する処理技術の開発であり，そのためには

① 斜線・曲線を含めた，画素形漢字パターンからのストロークの抽出法の検討

② 斜線・曲線の表現形式の検討

さらに，広く

③ 文字品質評価法の検討

が必要である。

また，データ圧縮法については，メモリ価格や，漢字パターンの変換技術の動向にもよるが，当面

④　大サイズの漢字パターンに対するデータ圧縮法の検討

が必要であると考える。

なお，本研究においては画素形漢字パターンを対象に考察したが，各種の変換処理に対する融通性を考えるとストローク形漢字パターン処理の研究は大きな課題であろう。

謝　　辞

本研究をまとめるにあたり，御指導，御助言をいただきました京都大学工学部，長尾真教授矢島脩三教授，また，御鞭撻を賜わりました池上淳一教授に謹んで感謝いたします。

本論文は日本電信電話公社研究所において行った研究をまとめたものであり，この間，御指導いただきました釜江尚彦画像通信研究室長，御鞭撻下さいました大森喬画像通信研究部長，伊吹公夫特別研究室長，山岸金吾日本通信技術株式会社通信システム開発部長，刑部稔技術局データ処理部門担当調査役，米沢進画像応用研究室長，広山昌生調査役，村上伸一調査役，谷口道夫調査役，若菜忠調査役に厚くお礼申し上げます。

また，本研の遂行を通して，討論とプログラミングで多大なる御協力をいただいた中野博隆調査員，名倉正計画像応用研究室長補佐，漢字パターンを提供していただいた小川克彦研究主任，ならびに画像応用研究室の皆様に心から感謝いたします。

参　考　文　献

（１）　情報交換用漢字符号系　ＪＩＳ．Ｃ６２２６，昭和５３年１月１日制定，日本規格協会発行

（２）　漢字入力装置の解説文献としては，例えば

（２－１）　長谷川実郎："漢字処理装置"，情報処理，Ｖｏｌ．１９，Ｎｏ．４，ｐｐ．３５０～３５８（１９７８）．

（２－２）　渡辺定久："漢字入力装置"，信学誌，Ｖｏｌ．６３，Ｎｏ．７，ｐｐ．７０７～７１２（１９８０）．

（３）　文字認識技術の概説記事としては，例えば

（３－１）　森健一："印刷文字の認識技術"信学誌，ｐｐ．１０７～１１５，Ｖｏｌ．６１，Ｎｏ．２（１９７８）．

（３－２）　安田，他："手書文字の認識"，信学誌，ｐｐ．１１５～１２４，Ｖｏｌ．６１，Ｎｏ．２（１９７８）．

（３－３）　増田功："日本語文字読取り装置"，信学誌，Ｖｏｌ．６３，Ｎｏ．７，ｐｐ．７１９～７２３（１９８０）．

（４）　例えば，

（４－１）　坂井，長尾，寺井："部分パターンによる漢字の合成"，情報処理，Ｖｏｌ．１０，Ｎｏ．５，ｐｐ．２８５～２９３（１９６９）．

（４－２）　禰津，大森："漢字パターンのデータ圧縮"，信学会電子計算機研究会資料，ＥＣ７０－５１（１９７１．３）．

（４－３）　黒崎悦明："高速漢字プリンタシステムについて"，情報処理，Ｖｏｌ．１６，Ｎｏ．９，ｐｐ．８０２～８０７（１９７５）．

（４－４）　谷口，森："漢字ドットパターンの圧縮の一考察"，信学技報，ＩＥ７３－５１（１９７３）．

（４－５）　森克己："漢字パターン処理"，（パターン生成，パターン変換，データ圧縮）"，昭５５電気四学会連合大会予稿，３０－４，（１９８０）．

（５）　長谷川実郎："パターン合成による漢字入出力処理"，情報処理，Ｖｏｌ．１６，Ｎｏ．９，ｐｐ．８０８～８１７（１９７５）．

（６） H.Ishida, S.Furukawa：" Synthesis and Display of Kanji by Unit Construction ", Preprint Seminer I/O System Japan Chinese Characters, Tokyo, pp.２７６～２８２（１９７１）.

（７） 吉田，小林："ストローク法による漢字パタンの一圧縮法"，昭４９信学全大予稿，No.１２９２（１９７４）.

（８） 新井，加藤，安田："ストローク法による漢字データ圧縮の一改良"，昭５０信学全大予稿，No.９７１（１９７５）.

（９） 新井，安田，加藤："画素形漢字データ圧縮の二，三の方法"，画像電子学会誌，第６巻，第１号，pp.１６～２４（１９７７）.

（10） 長谷川，真田，富永："漢字パターンにおけるデータ圧縮の一方法"，昭５２信学総全予稿，No.９７０（１９７７）.

（11） 木田，保坂，富永："漢字図形の発生方式"，昭５３信学総全予稿，No.１１１１（１９７８）.

（12） 安藤，阪口："文字パターン発生表示装置，１９７８年テレビジョン学会全国大会予稿，５－１２（１９７８）.

（13） 石田，永原，小西："漢字パターンデータの一圧縮方式について"，情報処理学会論文誌，Vol.２０，No.２，pp.９９～１０４（１９７９）.

（14） 祢津孔二："画素間の相互情報を利用した文字パターンの符号化法"，信学論，Vol.５５－D，No.４ pp.２７７～２８４（１９７２）.

（15） 古丸，嶋村，木村："画素形漢字パタンにおけるデータ圧縮の一方法"，昭５１信学総全予稿，No.９９９（１９７６）.

（16） 高木，津田，工藤："２次元予測による漢字パターンのデータ圧縮の検討"，昭５１信学総全予稿，No.１０００（１９７６）.

（17） 富田，岩田，上野，大西："漢字パターン圧縮の一方法"，昭５０信学全大予稿，No.９７２（１９７５）.

（18） S. Yajima, J.L.Goodsell, T.Ichida, H.Hiraishi：" DATA COMPRESSION OF KANJI CHARACTER PATTERNS DIGITIZED ON THE HEXAGONAL MESH", IEEE Trans. on Pattern Analysis and Machine Intelligence,Vol. PAMI-3 ,No.2, pp.221～230 (1981)

（19） 森克己："漢字ドットパタンのデータ圧縮に対する一考察"，昭５４信学総全予稿，

No．１０８７（１９７９）。

（20） 森克己：〝文字パターン伝送法に関する一考察〟，１９７９年テレビジョン学会全国大会予稿，１１－３（１９７９）。

（21） 佐橋，堀口：〝漢字パターンの伝送圧縮に関する一検討〟，昭５４信学総全予稿，No．１０９０（１９７９）。

（22） 佐橋，堀口：〝漢字パターンの伝送圧縮に関する一検討〟，信学技報，Vol．７９，No．１６，ＩＥ７９－５（１９７９．４）．

（23） 広山，谷口，堀口，服部：〝高速ファクシミリ応答方式－高速ファクシミリ受信機の漢字プリンタへの適用－〟，画像電子学会誌，Vol．６，No．２，pp．５８～６６（１９７７）．

（24） 谷口，森：〝漢字ドットパターンの圧縮の一考察〟，信学技報，ＩＥ７３－５１（１９７３）．

（25） 谷口，森：〝画素形漢字パターン圧縮の一方法〟，信学論，Vol．５７－Ｄ，No．６，pp．３７６～３８３（１９７４）。

（26） 原田威男：〝新聞用ディジタル・フォントの製作〟，印刷雑誌，Vol．６３，２，pp．５５～６０（１９８０）。

（27） 森克己：〝画素形漢字パターンのデータ圧縮に関する二，三の考察〟，信学論，Vol．６４－Ｄ，No．７，pp．６１７～６２４（１９８１）。

（28） 森克己：〝ドット漢字パターンマトリスの次数変換法〟，信学論，Vol．６０－Ｄ，１０（１９７７）。

（29） 中野，森：〝ローカルな性質を利用した漢字パターンの次数縮小変換〟，信学技法，ＩＥ７９－３（１９７９．４）。

（30） 渡部，筒井，加藤，及川，石山：〝漢字パターンの拡大・縮小法の一考察〟，信学技法，Vol．７９，No．１６，ＩＥ７９－２（１９７９．４）。

（31） 福来，及川，萬代，小島，斉藤，安西：〝漢字処理システムのハードウェア〟，日立評論，Vol．６０，No．５（１９７８－５）。

（32） 井上栄：〝漢字パターンのディジタル的補間拡大について〟，信学技報，ＥＣ７７－２６（１９７７）。

（33） 井上，木村，武川：〝漢字パターンの拡大・縮小法〟，信学技法，Vol．７９，No．１６，ＩＥ７９－１，（１９７９．４）。

(34) 大川，渡辺，風間，伊藤：〝漢字パターン拡大方式の検討〟，画像電子学会予稿，76－05－2（1976）。

(35) 斎藤，松葉，杉山：〝インクジェット式漢字プリンタ用画素形文字構成法〟，信学論，Vol．J62－D，No．11，pp．734～741（1979）。

(36) 斎藤，松葉，杉山：〝画素形文字パターンの次数変換法〟，昭53信学総全予稿，No．2041（1978）．

(37) 塩野，真田，手塚：〝漢字ドットパターンの次数変換と整形の一手法〟，信学論，Vol．J63－D，No．7，pp．557～564（1980）。

(38) 小川，中根：〝漢字パターン変換処理方式〟，信学技報，IE79－91（1980．2）。

(39) 小川，中根，池沢：〝漢字パターン変換のための差分データ圧縮法の一検討〟，昭56信学総全予稿，No．1029（1981）。

(40) 小川，中根，池沢：〝漢字パターン変換処理の一検討〟，信学論，投稿中。

(41) 塩野，真田，手塚：〝漢字パターンの明朝体からゴチック体への字体変換法〟，テレビジョン学会1981年全国大会予稿，16－7。

(42) 塩野，真田，手塚：〝明朝体漢字パターンからゴシック体漢字パターンへの変換法〟，昭56年度画像工学コンファレンス4－5（1981）。

(43) 百瀬，藤村：〝構造情報を利用した明朝体漢字パターンの発生法〟，第11回画像工学コンファレンス。

(44) 吹抜敬彦：〝漢字パターンのデータ圧縮の一方式－Modified Stroke 方式－〟，昭53信学総全予稿，No．1109（1978）。

(45) 小田，福森，金出：〝マン・マシン対話方式によるドット文字，ドットパターン作成方式とその実施例〟，情報処理，Vol．16，No．8，pp．685～691（1975）。

(46) 小畑，風間，伊藤：〝漢字パターン処理システム〟，昭51信学総全予稿，No．1133（1976）。

(47) 松林，竪月：〝漢字・図形作成管理システム〟，昭56信学総全予稿，No．1204（1981）。

(48) 小谷，大迫，安孫子：〝各種漢字パターン・メモリの実例とその得失〟，日経エレクトロニクス，pp．126～148，（1978．2．20）。

(49) 例えば，
"マスクROM大容量・高速化進む"，電波新聞，1981年5月25日号
(50) 原島，真柄："画像応答システム（VRS）"，テレビジョン学会誌，第34巻，第10号，pp.909～912（1980）。
(51) 広山，谷口，堀口："ファクシミリ受信機の漢字プリンタへの適用"，画像電子学会予稿74－01－4（1974）。
(52) 磯崎，大越："キャプテンシステム"，信学誌，Vol.64，No.8，pp.834～840（1981）。
(53) Katumi MORI, Masakazu NAGURA："A Data Compressing Method for Dot Matrix Patterns of "Kanji" Characters"，Review of the Electrical Communication Laboratories，Vol.24，Nos.1－2，pp.106－114（1976）。
(54) 禰津孔二："文字パターンのデータ圧縮のためのパターン変換方法"，昭47信学全大予稿，No.1790（1972）。
(55) 小田，能勢："漢字イメージデータの圧縮"，昭和48年度情報処理学会第14回大会予稿，No.158（1973）。
(56) 佐藤，鈴木，下村："漢字パターン圧縮記憶の一方法"，昭49信学全大予稿，No.1291（1974）。
(57) 野村，小池："日本語文字発生方式"，情報処理，Vol.21，No.11，pp.1136～1142（1980）。
(58) 森，名倉："画素形文字パタンデータ圧縮法"，研究実用化報告，第24巻，第5号pp.927～940（1975）
(59) THOMASS.HUANG："Coding of Two－Tone Images"，IEEE TRANS.ON COMMUNICATIONS，VoL.COM－25，No.11，pp.1406～1424（1977）。
(60) 若原，山崎，寺村，中込："変化点相対アドレス符号化によるファクシミリ信号のデータ圧縮"，信学技報，CS74－115（1974－11）。
(61) TOYOMICHI YAMADA："Edge－Difference Coding－A New，Efficient Redundancy Reduction Technique for Facsimile Signals"，IEEE TRANS. Vol. COM－27，No.8，pp.1210～1217（1979）。

(62) 結城，山田，山崎，若原：“ファクシミリ信号のＲＥＡＤ符号化方式”，画像電子学会研究会，７８－０６－４（１９７９．２．１３）。

(63) 森口，宇田川，一松著，数学公式Ⅱ，岩波全書２２９，１９５７年第１刷発行。

(64) 活字における大きさ，書体の解説書としては，例えば，
佐藤敬之輔：“日本のタイポグラフィ”，紀伊国屋書店，１９７２年８月第１刷発行。

(65) 中野，森，釜江：“テーブルを用いた漢字パタンの次数縮小変換法について”，昭５３信学会通信部門全大予稿，No．５１０（１９７８）。

(66) 井面，岡野，高田，佐伯，田辺：“ドット化文字のスムージング拡大法”，１９７８年テレビジョン学会全国大会予稿，５－１３（１９７８）。

(67) 塩野，真田，手塚：“曲面補間による漢字ドットパターンの次数変換”，信学技報，ＩＥ７８－７２（１９７８）。

(68) 渡辺，風間，伊藤，亀山：“文字パターン拡大方式に関する一検討”，昭５２信学総全予稿，No．９７２（１９７７）。

(69) 清水，吉田：“漢字パターンの縮小方式”，昭５６信学総全予稿，No．１４０３（１９８１）。

(70) 長尾真　監訳“ディジタル画像処理”，近代科学社，昭和５３年１２月１０日発行。

(71) 森，大山：“ドット文字パターンの拡大・縮小法の一考察”，昭５１信学総全予稿，No．９６５（１９７６）。

(72) 森克己：“漢字ドットマトリックスの次数変換法”，信学技報，ＩＥ７６－８１（１９７７．２）

(73) 中野，中田：“周辺分布とそのスペクトルによる漢字の認識”，信学論，Vol．５６－Ｄ，No．３，pp．１４６～１５３（１９７３）。

(74) 森克己：“出力技術としての漢字パターン処理”，情報処理，Vol．２２，No．７，pp．６５０～６５６（１９８１）。

(75) 南，富永：“漢字パターン発生方式におけるストローク情報の拡張”，昭５３信学会通信部門全大予稿，No．５１１（１９７８）。

(76) 保坂，富永：“境界線生成による漢字図形の発生”，昭５３信学会通信部門全大予稿，No．５１２（１９７８）。

(77) 例えば，
山本，宮本，安田，提：“手書数字認識論理の設計”，信学論，Vol．５３－Ｃ，

No．１０，p．６９１（１９７５）。

（78） 中野，森：“漢字パタンにおけるストローク情報抽出法に関する一考察”，昭５４信学総全予稿，No．1088(1979)。

（79） 中野，森：“直線分離法による漢字パターンの処理”，信学論，Vol．６２－D，No．１２（１９７９）。

（80） 森，中野：“漢字ドットパタンの作成処理”，情報処理学会イメージプロセッシング１７－１（１９７８．３．２２）

（81） K. Mori, H.Nakano：“COMPUTER AIDED DESIGN OF DOT MATRICES FOR KANJI CHARACTERS”, Ⅳ th IJCPR, pp．８２９～８３１（１９７８）。